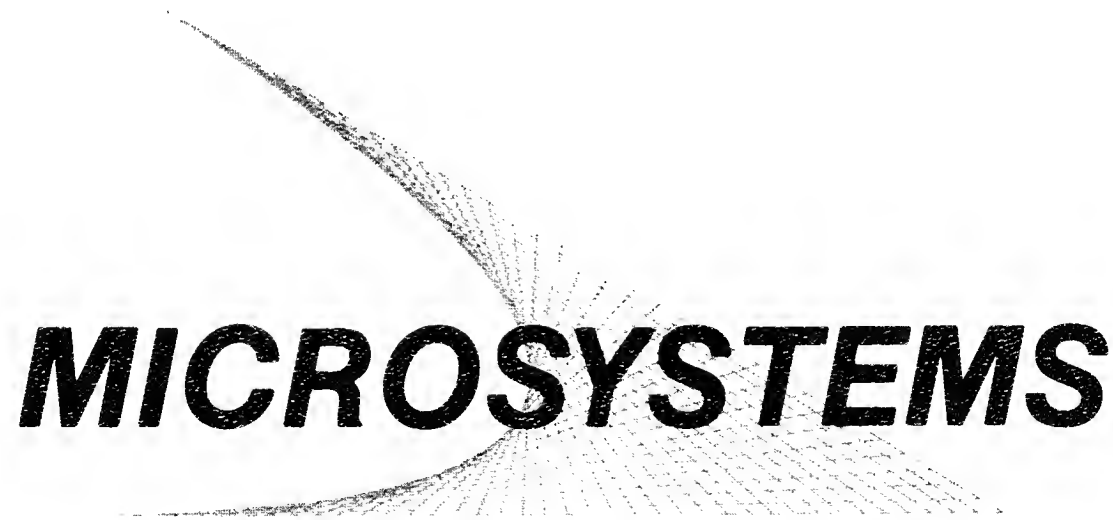
MOTOROLA

RMS68KIO/D8

VERSAdos
Data Management Services
And Program Loader
User's Manual



QUALITY • PEOPLE • PERFORMANCE

MOTOROLA

RMS68KIO/D8

JANUARY 1986

VERSAdos

DATA MANAGEMENT SERVICES

AND PROGRAM LOADER

USER'S MANUAL

Eighth Edition

Seventh Edition March 1985

MICROSYSTEMS

REVISION RECORD

RMS68KIO/D7 -- March 1985. Reflects the VERSAdos 4.4 software level. Adds support of the MC68020, VM04, VME/12, MVME120, MVME121, MVME122, MVME123, and various VERSAdos drivers.

RMS68KIO/D8 -- January 1986. Revised for clarity and consistency. Added keyword index.

TABLE OF CONTENTS

TABLE OF CONTENTS (cont'd)

TABLE OF CONTENTS (cont'd)

TABLE OF CONTENTS (cont'd)

TABLE OF CONTENTS (cont'd)

Page

LIST OF ILLUSTRATIONS

LIST OF TABLES

THIS PAGE INTENTIONALLY LEFT BLANK.

1

CHAPTER 1

DATA MANAGEMENT FACILITIES

1.1 INTRODUCTION

1.1.1 Manual Contents

This manual contains information on data management and program loading facilities available in the VERSAdos operating system.

Chapters 1, 2, 3, and 4 describe:

Device names and file descriptors
Disk, file, and record structures
File access methods
File and device protection
Temporary and spooler files
File Handling Services (FHS) module
Input/Output Services (IOS) module
Supported devices

Chapter 5, "PROGRAM LOADER", shows how any task can call the loader to transfer a load module into memory and create a new task. The parameter block required for calling the loader, by executing a TRAP #4, is also described.

1.1.2 Notation

In this manual, commands and other Input/Output (I/O) are presented in a modified Backus-Naur Form (BNF). Certain symbols in the syntax may be used where noted in the real I/O; however, others are meta-symbols whose meanings are:

< > Angle brackets enclose a word referred to as a syntactic variable that is replaced in a command line by one of a class of items it represents.

[ ] Square brackets enclose an item that is optional. The enclosed item may occur zero or one time.

[ ]... Square brackets followed by periods enclose an item that is optional/repetitive. The item may appear zero or more times.

Operator inputs are followed by a carriage return. The carriage return is shown as (CR) if it is the only input required. In some examples, operator inputs are underscored for clarity. The underscore is not typed.

1

1.1.3 Related Documentation

The following publications may provide additional helpful information. If not shipped with this product, they may be obtained from Motorola's Literature Distribution Center, 616 West 24th Street, Tempe, Az 85282; telephone (602) 994-6561.

DOCUMENT TITLE	MOTOROLA PUBLICATION NUMBER
M68000 CRT Text Editor User's Manual	M68KEDIT
M68000 Family Real-Time Multitasking Software User's Manual	M68KRMS68K
System Generation Facility User's Manual	M68KSYSGEN
VERSAdos Messages Reference Manual	M68KVMSG
VERSAdos Overview	M68KVOVER
M68000 Family VERSAdos System Facilities Reference Manual	M68KVSF
VERSAdos to VME Hardware and Software Configuration User's Manual	MVMEVDOS
Guide to Writing Device Drivers for VERSAdos	M68KDRVGD

1.2 INPUT/OUTPUT SERVICES OVERVIEW

Because the VERSAdos I/O system handles logical I/O rather than physical hardware I/O, the VERSAdos system I/O is device independent. Most I/O operations refer only to logical properties (e.g., the next record), rather than to particular device characteristics or file formats. VERSAdos-supported hardware I/O is performed by routines not directly accessible to a user task.

To help control the sources and targets for I/O, the system uses a software feature called a logical unit assignment. An assignment is similar to a numbered channel; it controls the flow of data between program accessible storage and devices/files. Before any I/O operations can occur, file assignment must be made. The assignment specifies a Logical Unit Number (LUN) and target (file or device) as well as the type of channel connection required (e.g., exclusive read and write). Paragraph 2.3.2 describes the Assignment facility in detail.

A file's assignment type governs which I/O calls are allowed. For example, to rename a file requires an Exclusive Read-Write (EREW) assignment.

All I/O is separated into two categories:

IOS call Executed via a TRAP #2 instruction.

FHS call Executed via a TRAP #3 instruction.

FILE HANDLING SERVICES (FHS)

Allocate	Create a file on a random access device
Assign	Establish a logical connection
Change-Access-Permission	Change access permission of a file or device
Rename	Change a file ID
Protect	Change access privileges
Delete	Delete a file
Checkpoint	Update a file on disk
Close	Dissolve an assignment
Retrieve-Attributes	Get device/file attributes
Fetch-Directory-Entry	Get directory entries
Fetch-Device-Mnemonics	Get device/volume IDs
Change-LUN-Assignment	Switch LUN assignment
Fetch-Default-Volume	Retrieve system or user default volume
Cancel	Cancel a spooler job
Continue	Continue a spooler output
Forms	Change forms ID
Print	Add/change a spooler job
Copies	Change number of copies for a spooler job
Queue	Output the spooler queue listing

INPUT/OUTPUT SERVICES (IOS)

Read	Read from a file/device
Write	Write to a file/device
Connection-Wait-I/O	Suspend task until device is free
Proceed-Wait-I/O	Return control to the calling task
Output-with-Input	Write to then read from a device
Update-Record	Update a record
Delete-Record	Delete a record
Format	Format a disk/sector
Position	Position to a particular record of a file
Rewind	Position at beginning of file
Halt-I/O	Terminate outstanding I/O
Test-I/O-Complete	Test I/O completion
Wait-Only	Wait only
Local-Break-Claimer	Request an attention event for a break condition
Transmit-Break	Send break to LUN specified
Negate-Local-Break-Claimer	Release from break service
Configure-Device	Set device-specific configuration
Configure-Defaults	Change device-specific parameters assigned at SYSGEN
Configuration-Status	Return device configuration

1

The FHS module provides the logical link to a file via a Logical Unit Number (LUN). The FHS module is also called whenever a logical link is changed or dissolved. In addition, the FHS module is called to create a disk file and whenever the target attributes need to be changed (i.e., change access permission) for file or device assignments.

All data transfers are handled by calling the IOS module. The logical unit connection between the task and the physical device must have been made with the Assign (FHS) request before invoking IOS with a TRAP #2 (refer to paragraph 2.3.2).

1.3 FILE AND DEVICE NAME SPECIFICATION

A VERSAdos disk file description consists of five fields:

a. Volume ID — A string of one to four alphanumeric characters, the first of which must be alphabetic. It is the name of the volume on which the file resides.

b. User number — A 2-byte binary number with a value of:

1 to 65533 Indicates a private user file.

0 Indicates a system file.

Paragraph 2.2.5 describes the user number concept.

c. Catalog name — A string of one to eight alphanumeric characters, the first of which must be alphabetic. Any spaces must be trailing spaces. A null catalog (all spaces) is a valid catalog name. Refer to paragraph 2.2.6.

d. Filename — A string of one to eight alphanumeric characters, the first of which must be alphabetic. Any spaces must be trailing spaces. This is the main identifier for the file and the choice of name is at the discretion of the user. Refer to paragraph 2.2.7.

e. Extension — A string of one to two alphanumeric characters, the first of which must be alphabetic. The extension usually denotes the type of data in the file and can be anything the user chooses. However, the VERSAdos operating system and utilities use some default extensions:

AF	Chain files (assembly of source modules)
AG	Source files (assembled at SYSGEN)
AI	INCLUDE files (not assembled at SYSGEN)
CD	SYSGEN command files
CF	Chain file
CI	Included SYSGEN command file
DB	SYMbug debug files from linker
EM	Emulator module files for SYMbug/A and HDS
EQ	Equate files
FF	FORTRAN compilation chain files
HT	Copyright and Restricted Rights Legend files
IN	Initialization files
LF	Chain files for linking RO modules
LG	Link chain files (run at SYSGEN)
LL	Output listing files from a linkage
LO	Load modules output from linking
LS	Output listing files from an assembly
MC	INCLUDE files (assembler MACRO source)
MG	Merge files (insert into existing files)
MX	S-Record format files
NW	News files
OW	Owner files (session management)
PC	Pascal intermediate code files
PF	Pascal compilation (chain or profile files)
PL	Pascal phase 1 listing files
PO	Optimized Pascal code files
RO	Relocatable object modules
RS	Symbolic debug files (from ASM for linker)
SA	Source ASCII files
SI	INCLUDE files (assemblies done at SYSGEN)
SY	System related files
TF	Temporary files

Other extensions are defined as required.

File descriptors are written as:

```
<voln>:<user>.<catalog>.<filename>.<ext>
```

where:

<voln>	is the volume ID.
<user>	is the user number.
<catalog>	is the catalog name.
<filename>	is the filename.
<ext>	is the extension.

A file description can also describe a device, in which case <voln> describes a device mnemonic rather than a volume ID. At initialization time, each device in the system is assigned a device mnemonic of up to four characters.

The format for specifying a device mnemonic is:

```
# <dev>
```

1

No volume ID may match any device mnemonic within a single system. Each device mnemonic defined for a single system must be unique.

For a complete description of command level syntax, refer to the M68000 Family VERSAdos System Facilities Reference Manual.

1.3.1 Default Values

FHS handles default values for the system volume ID or user default volume ID and user number only. The session manager handles other defaults. (Refer to the M68000 Family VERSAdos System Facilities Reference Manual.) The volume ID has several default values, depending on the type of file specified. To indicate that the default should be used, set the volume ID field to spaces ($20).

The volume ID system default value is established when the system administrator initializes the system. When user session management creates a session at user logon, user default values are established for volume ID, user number, and catalog.

The user number must be specified at user logon; volume ID and catalog may also be specified. If volume ID and catalog are not specified, the system volume ID is used as the default user volume ID, and the default user catalog is set to all spaces. A user can change the user default values during a session with the USE command.

When the system administrator (user number = 0) omits the volume ID in specifying a file, the volume ID first defaults to the default user volume, then to the default system volume. When specifying a user file (which has user number $\neq$ 0 and does not reside on the system volume), a default volume ID must have been named (or changed with the USE command) at logon. If specifying a system file (user number = 0), the volume ID may be omitted; the system volume is the default value.

In addition to a system volume for system files and user volumes for user files, a temporary file volume and a spooler file volume are used to contain temporary files and spooler files, respectively. An ampersand (&) as the first character in the filename field identifies a temporary file. A commercial at sign (@) in the first character position of the filename field defines a spooler file.

When the spooler task is created, the system administrator may provide a spooler volume ID. If no volume ID is specified at that time, the system volume ID is the default value.

The default temporary volume ID always uses the system volume ID. When the LUN assignment between the file and the temporary volume is closed, temporary files are deleted.

If the user number field value is -1, it defaults to the user's logon user number. A value of -2 defaults to all user numbers only for a Fetch-Directory-Entry request (refer to paragraph 2.3.10).

1.3.2 Family Names

Family names apply only to random access files and mean that a group of files have a common element in their file descriptors. Some FHS commands support handling of family names; for example, the Protect command can be used to protect all files on volume DSK1 with the filename of PASCAL and any extension.

FHS only recognizes a family on the Fetch-Directory-Entry request. An asterisk (*) in any position of the catalog, filename, or extension field means match any character (e.g., A*******.SA means any filename beginning with A and ending with extension SA). A user number of -2 returns files for all user numbers.

1.3.3 User Number Protection

The user validation file identifies each authorized user and is accessible only to the system administrator. To use the system, a user must log on to the system; this could involve a password known only by the user and the system administrator.

To coordinate a group of users, the file manager recognizes the concept of private and system files. System files are readable by all users but can only be written by either the system administrator or a system task. Private files are readable by all users (provided the matching read-protect code is supplied or the file-protect code is 0), but can only be written to by the owner or system administrator. The file system uses the user number of the authorized user to distinguish among private filenames; thus, users can freely use any private filename without interference with other user files.

1.4 DISK STRUCTURE

The physical layouts of data on hard disk and floppy disks are not the same. The two devices have different sector sizes, a different number of sectors per track, and a different number of tracks per disk. These differences are normally transparent to the user because the respective disk controller handles them. Sectors are accessed on either device via a Physical Sector Number (PSN). The corresponding disk controller decodes the PSN into the appropriate cylinder/sector position. (When referring to "sector" in this manual it is defined as a logical sector that is always 256 contiguous bytes of data.)

A portion of each disk (both hard and floppy) is reserved for some special system tables. Although the tables may not be the same size, they are identical in format between the two disk devices.

For additional details on disk data structures, refer to the description of the REPAIR utility in the M68000 Family VERSAdos System Facilities Reference Manual.

1.4.1 Volume Identification Block (VID)

The Volume Identification Block (VID) is created during disk initialization. It contains a volume ID, the version and revision number of the resident operating system, the date the disk was generated, a user name identification area, and a user number. In addition, it contains pointers to the Sector Allocation Table (SAT) and directory. The VID is the only system table that must always reside in the same place on the disk, i.e., at PSN 0. Refer to the description of the INIT utility in the M68000 Family VERSAdos System Facilities Reference Manual for an explanation of the creation of the VID.

1.4.2 Configuration Area (CFGA)

Disk initialization creates the Configuration Area (CFGA) that contains information about the configuration in effect at the time the disk is initialized. This information is used to set the current configuration at the time the disk is mounted. Refer to the description of the REPAIR utility in the M68000 Family VERSAdos Systems Facilities Reference Manual and the disk configuration parameter block (refer to paragraph 3.5.5) in this manual for details on the contents of the CFGA.

1.4.3 Sector Allocation Table (SAT)

The SAT contains a bit map of the areas on the disk that are available for new space allocation. Each bit in the SAT represents a sector of disk storage. A bit set to 1 indicates that the sector is allocated. If a bit is set to 0, it indicates that the corresponding sector is available for allocation. The parts of the SAT that represent sectors beyond the physical end of the disk are marked as allocated so that they cannot be used. The start of the SAT and the number of sectors is in the VID. The number of sectors allowed for the device determines the length.

1.4.4 Primary Directory Block

A directory is maintained on each volume and contains the names of the files residing on that volume.

The information maintained in the directory entry for each file is:

- Filename
- Extension
- File start and length or File Access Block (FAB) pointer
- File type
- Write and read access codes
- Record size

The directory is expandable, with its size limited only by the amount of available disk space.

1.4.5 Secondary Directory Block

A secondary directory block is also maintained and contains an entry for each unique user number/catalog on the volume.

1.5 FILE STRUCTURE

While the contents of a file can be thought of as a logically contiguous block of information, the actual disk area allocated to the file may or may not be physically contiguous. Space can be allocated to one or more groups of contiguous sectors on the disk. Each contiguous group of sectors is called a segment. This segmentation allows the dynamic allocation and de-allocation of space to occur without having to move any of the information contained in the file or in other files.

Except for contiguous files, each file must have a table describing which segments are allocated to the file. This is referred to as the File Access Block (FAB).

The I/O service routines access sectors within a file by Logical Sector Number (LSN). LSNs for data sectors are numbered sequentially beginning with 0. Thus, even though a file may be segmented (not physically contiguous on the disk), it is treated as a logically contiguous collection of sectors when accessed by an LSN. The system I/O functions decode the LSN into the PSN.

1.5.1 File Formats

Both hard and floppy disks support four file formats:

Contiguous
Sequential
Indexed sequential with duplicate keys
Indexed sequential without duplicate keys

Contiguous files include binary load module files whose contents are to be loaded into memory directly from the disk, in addition to any other type of user-defined data that must have contiguous allocation. Contiguous files are allocated with the maximum amount of space they will ever need. The only information retained in binary load module files about where to load the content is kept in the first sector of the file called the Loader Information Block (LIB). The other data within the remaining file sectors contains no load or record information; it is just an image of a memory block to load.

Sequential and indexed sequential files do not necessarily have contiguous allocation. Each has a FAB describing where all the segments of the file reside. Sequential file records may be accessed sequentially or randomly by logical record number. Indexed sequential file records can be accessed sequentially or randomly by key or logical record number.

1

1.6 RECORD STRUCTURE

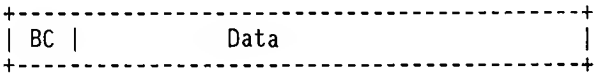
This section describes the three record types supported for disk files. Chapter 4 discusses the use of these records and the supported I/O functions.

1.6.1 Contiguous Records

Contiguous file records are also referred to as blocks. Each block is 256 bytes in length. The user specifies the complete data content of the block. There are no embedded control characters generated by the File Management Service (FMS). (FMS is the facility that performs the FHS and IOS functions requested for files.)

1.6.2 Variable Length Records

The format of variable length records is:



where:

BC = A word containing the number of bytes of data in the record.

Data = The maximum number of data bytes is 65,280 ($FF00). The data portion of the record can be any binary data.

With the ASCII specification and a formatted write request, FMS compresses spaces. A data byte having the sign bit (bit 7) set to 1 indicates a space compression character. The remaining bits (0 to 6) contain a binary number representing the number of spaces to insert in place of the compressed character. FMS automatically expands these characters into spaces when such files are read using formatted ASCII I/O. When the file is created (allocated), if the specified fixed record length is 0, it has variable length records. If the record length is odd, a 0 filler byte trails the end of the data portion.

1.6.3 Fixed Length Records

The user specifies the complete data content of the fixed length records. There are no embedded control characters generated by FMS. If, on allocation, a nonzero fixed record length was specified (i.e., when the file was created), sequential and indexed sequential files have fixed length records.

1.7 SYSTEM FILES

At initialization, the file named ERRORMSG.SY is required. If not present, an error message is displayed after the boot process is complete.

The user validation file contains a list of valid user numbers. Each user number has an associated password. The specified user number and password must be an entry in this file to enable the user to logon to the system.

The user validation file is in indexed sequential format and allows user numbers from 0 to 9,999. At least one entry must always exist in the file; an entry containing a user number of 0 is always required.

1.8 FILE ACCESS METHODS

VERSAdos supports two methods of access to files:

a. Random
b. Sequential

For sequential access, three modes exist: Next, Current, and Prior. Random and sequential access may be intermixed without closing and reassigning the file.

A current record pointer is maintained for each assigned file that indicates the record to be read or written on the Next sequential access. At the time of assignment, the current record pointer is set so that a Next request accesses the first record in the file. If the assignment was for position at end, the current record pointer is set so that records can be appended to the file by writing Next.

On completion of either a random or sequential request, the current record pointer is set so that a request for Current accesses the same record; a request for Next accesses the next record in sequence (unless it is a read request and no new records exist); and a request for Prior accesses the prior record (unless the current record pointer is already at the beginning of the file).

A Rewind I/O request sets the pointer so that a Next request accesses the first record in the file. These commands set the current record pointer similar to an I/O Position request.

1.8.1 Random Access

For random access, the user supplies the record number to be accessed. The record is found, the data transfer is performed, and the current record pointer is set to point to the random record requested. If the user continues to use random access, the current record pointer is ignored because it is readjusted on every call. However, the user may read or write a sequence of records starting with a known record number. Here, a single random call may be followed by a number of sequential calls for Next.

With a sequential file, the user is somewhat restricted in using the random access Write and Update-Record calls. The Update-Record call may be used to replace any record of the same length currently in the file. The Write call may be used to append one record to the end of the file. If the record number specified is more than one record past the end of the file, the call is rejected with End-Of-File (EOF) status, which means a file must be expanded in a sequential manner. For example, if the file has only five records, a sixth may be added, but a record number 15 could not be added. If an Update-Record call requests a non-existent record, an EOF error is returned.

With an indexed sequential file, the user may access the file randomly in two ways:

a. By logical record number (identical to sequential file access).

b. By key value.

If the file access is by key value, an Update-Record, Read, or Delete-Record request accesses the record with the requested key. For a Write request, the record is inserted in the file, positioned according to keys so that the keys remain in ascending order.

For a contiguous file, any record within the file allocation may be read or written in any order.

Noncontiguous files (sequential and indexed sequential) support record or block random I/O. To copy these types of files using record I/O would be time consuming; therefore, random logical sector I/O is supported for speed (i.e., a read or write of one sector might mean several records). Chapter 4 discusses this assignment mechanism.

1.8.2 Sequential Access

Sequential access is the simplest and most common access method. The user performs a series of sequential Read or Write calls, usually NEXT causing records of the file to be read or written in sequence. The current record pointer is automatically adjusted at each access.

1.9 FILE AND DEVICE PROTECTION

Files and devices have fixed and active protection.

1.9.1 Fixed Protection

Each file or device has two access protection codes associated with it: one for read access and one for write access. Each code is one byte long and has any value from $00 to $FF.

If the values of the protect codes are between $01 to $FE, the file or device may not be assigned for read or write access unless the operator or requesting task supplies the matching codes.

If a protect code value is $00, the file or device is unprotected for that access mode and any protect code is valid.

If a protect code value is $FF, the file is unconditionally protected for that mode. It may not be assigned to any non-owner user task for that access mode, regardless of the code supplied. The owner may assign a file with either a read- or write-protect code of $FF and read a protected file. However, the owner cannot write into a write-protected file. The purpose of assigning it was probably to change the write-protect code to later allow write access of the file. An unconditionally protected file may be assigned to an Executive task, including the system administrator.

Examples of fixed protection:

WRITE PROTECT CODE	READ PROTECT CODE	MEANING
$00	$00	Completely unprotected.
$FF	$FF	Unconditionally protected.
$0F	$00	Unprotected for read; conditionally protected for write (user must supply write code = $0F).
$FF	$0F	Unconditionally protected for write. Conditionally protected for read.
$00	$FF	Unprotected for write; unconditionally protected for read.
$32	$70	Conditionally protected for both read and write.

1

The file protection codes are defined when the file is allocated. The terminal operator or any task having that file assigned for exclusive access may change the access protection codes (refer to paragraph 1.9.2). The protection codes are changed with an operator Rename command or a Protect FHS call.

The device access protection codes are defined at initialization time; only the system administrator can change them.

1.9.2 Active Protection

When a task has assigned a file and it is being used, it can prevent other tasks from accessing that file. For this reason, the user may ask for exclusive access permission, for either read or write, at assignment time. This form of protection is called active because it is only in effect while the file remains assigned.

The access permission is known by the abbreviations:

PR	Public read
ER	Exclusive read
PW	Public write
EW	Exclusive write
PRPW	Public read-write
PREW	Public read, exclusive write
ERPW	Exclusive read, public write
EREW	Exclusive read-write

A file cannot be assigned with an access permission that is incompatible with an existing assignment for that file. For example, a request to open a file for exclusive write only is compatible with an existing assignment of that file for PR or ER but is incompatible with any existing assignment for other access permissions. Table 1-1 shows compatibilities and incompatibilities among access permissions.

1.9.3 Write Protection

When a volume is write-protected, only read assignments (public read or exclusive read) are accepted. If the hardware write-protected feature of a disk or tape is enabled, the volume is write-protected.

TABLE 1-1. Compatible Access Permission

	PR	ER	PW	EW	PRPW	PREW	ERPW	EREW
Public read	X		X	X	X	X		
Exclusive read			X	X				
Public write	X*	X*	X*		X*		X*	
Exclusive write	X*	X*						
Public read-write	X*		X*		X*			
Public read, exclusive write	X*							
Exclusive read, public write			X*					
Exclusive read-write								

* For sequential files only

1.9.4 Protection Modification

The terminal operator or any task having that file assigned for exclusive access can change a file's protection codes. If the task has the file assigned for exclusive read, it can change the read access protection code; if it is assigned for exclusive write, the write-protect code can be changed; and if the task has the file assigned for EREW, it can change either or both codes.

If the proper conditions are met, a task can change its file access permission without closing the file. For example, a task with a file assigned for public read cannot change to exclusive read if the file is also assigned for public read to another task (or another logical unit of the same task). Access can always be changed from exclusive to public. The Change-Access-Permission option of the FHS call performs this change.

If the user attempts to change access permission and is unable to get the new permission, the old access permission remains.

1.9.5 Multiple Logical Unit Considerations

Exclusive access has been discussed in terms of multiple tasks sharing the same file. It was assumed that a single task does not attempt to assign the same file to multiple logical units. However, sometimes the task wants to assign the same file to multiple LUNs occurring as a result of default assignments or the terminal operator's assignments. Here, exclusivity applies between LUNs just as between tasks, which means a file cannot be assigned for exclusive read access on one LUN and shared read on another. If a file is assigned for exclusive read or write access on any given LUN, it may not be assigned for that access on any other LUN.

1.10 TEMPORARY AND SPOOLER FILES

1.10.1 Temporary Files

Bulk storage of temporary data use temporary files. The temporary file operator command or a special form of the FHS allocates and assigns temporary files. Refer to paragraph 2.3.2.1.

Temporary files have a special filename consisting of the special character & and an alphanumeric string of characters based on the time and date of allocation.

Temporary files exist only as long as they are assigned, and are deleted when the assignment is closed. If a volume is not specified, temporary files are allocated on the default temporary volume. The set default volume command, DEF, can change the default temporary volume.

1.10.2 Spooler Files

Spooler filenames start with the special character @. When the file is allocated, FHS generates the remainder of the filename. Spooler files must be sequential. All requested record I/O is forced to be image. On record Write requests, the IOS parameter block options word is inserted at the beginning of the record. Rn read requests, the options word written with the record is returned as the first part of the record.

If a volume is not specified, spooler files are allocated on the default spooler volume. The DEF command can change the default spooler volume.

CHAPTER 2

FHS - FILE HANDLING SERVICES

2.1 GENERAL

A File Handling Services (FHS) call facilitates file and device manipulation. The facilities provided are:

a. Allocation (creation) of direct-access files
b. Assignment of files and devices to Logical Units (LUNs)
c. Modification of access permission on existing assignments
d. Renaming of files
e. Modification of file access protection codes
f. Closing (unassignment) of assigned files and devices
g. Deletion of direct-access files
h. Checkpointing of assigned files
i. Examination of attributes of assigned files and devices
j. Getting directory entries
k. Getting device mnemonics
l. Changing LUN assignment
m. Retrieving system or user default volume
n. Cancelling a spooler job
o. Continuing a spooler output
p. Changing forms ID
q. Adding or changing a spooler job
r. Changing the number of copies for a spooler job
s. Outputting the spooler queue listing

2.2 FHS PARAMETER BLOCK

The 40-byte FHS parameter block is described in Figure 2-1. Although all fields of this parameter block are not required for each FHS function, the full parameter block of 40 bytes must be reserved for all calls. The meaning and use of each field is explained in the description of each function requiring that field. The parameter block address is passed to FHS in register A0.

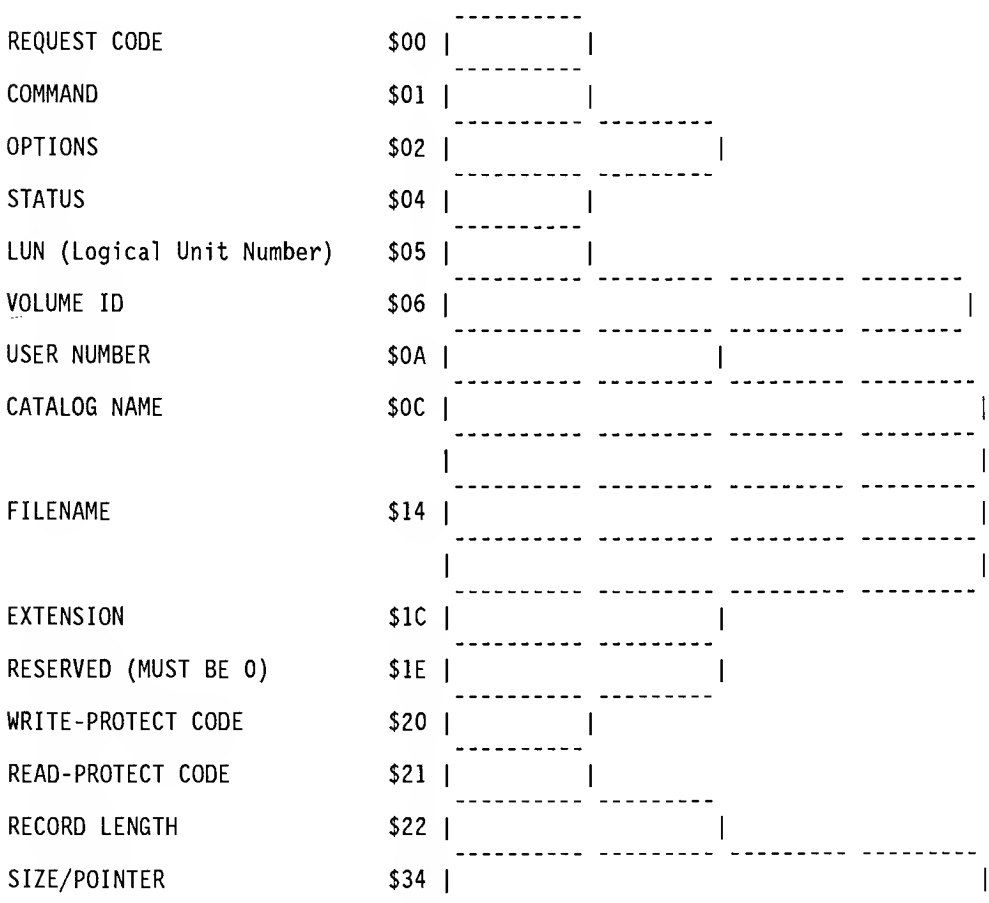


FIGURE 2-1. FHS Parameter Block (TRAP #3)

The parameter block must be on a word boundary, reside in a single memory management unit segment and may be coded as:

```
PARBLK:     DC.B    CODE        Command code
            DC.B    CMD         Command (1 byte)
            DC.W    OPT         Options (2 bytes)
            DC.B    0           Status (1 byte)
            DC.B    LUN         LUN (1 byte)
            DC.B    'VOLN'      Volume ID (4 bytes)
            DC.W    0           User number (2 bytes)
            DC.B    'CATALOG '  Catalog name (8 bytes)
            DC.B    'FILENAME'  Filename (8 bytes)
            DC.B    'EX'        Extension (2 bytes)
            DC.W    0           Reserved (2 bytes)
            DC.B    WCODE       Write code (1 byte)
            DC.B    RCODE       Read code (1 byte)
            DC.W    RECL        Record length (2 bytes)
            DC.L    SIZE        Size (4 bytes)
```

Appendix A provides a table specifying the required parameter for each function and the FHS parameter block.

The following paragraphs describe the FHS parameter block fields.

2.2.1 Commands/Options

The code field defines the bit assignments in the command field:

```
  $00       Device/file commands
  $01       Utility commands
  $02       Spooler commands
$03-$7F     Not used
  $80       Privileged use
$81-$FF     Not used
```

2.2.1.1 Code $00 Commands. If more than one command bit is set for code $00 commands, the FHS functions are sequentially processed from left to right.

BIT	MEANING	
7	Allocate	Requires file type field as option.
6	Assign	Requires access permission and transfer method fields as options.
5	Change-Access-Permission	Requires access permission field as option.
4	Rename	
3	Protect	
2	Close	
1	Delete	
0	Checkpoint	

2.2.1.2 Code $00 Options. The option code is defined as:

Bits 15-12 User-specified attributes.

Bit 11 Specifies the option to return the physical address of the shared data buffer. When bits 11 and 5 are set, the returned shared segment logical address is the same as the segment physical address. Refer to paragraph 2.2.13 for additional information.

Bits 10-8 File type:

000 = Contiguous
001 = Sequential
010 = Indexed sequential (no duplicate keys)
011 = Indexed sequential (duplicate keys allowed)
1XX = Reserved

Bit 7 Reserved.

Bit 6 Position option:

0 = Position current record pointer at beginning of file at assign

1 = Position current record pointer at end of file at assign

Bit 5 Shared segment option:

1 = Allocate shared data buffer at assign

Paragraph 2.2.13 describes additional parameters required if bit 5 is set.

Bit 4 Reserved.

Bit 3 File overwrite option:

1 = Overwrite existing file

Bits 2-0 Access permission:

000 = Public read (PR)
001 = Exclusive read (ER)
010 = Public write (PW)
011 = Exclusive write (EW)
100 = Public read-write (PRPW)
(Uses: device, output with input; file, normal read/write)
101 = Public read, exclusive write (PREW)
110 = Exclusive read, public write (ERPW)
111 = Exclusive read-write (EREW)
(Uses: device, output with input; file, normal read/write)

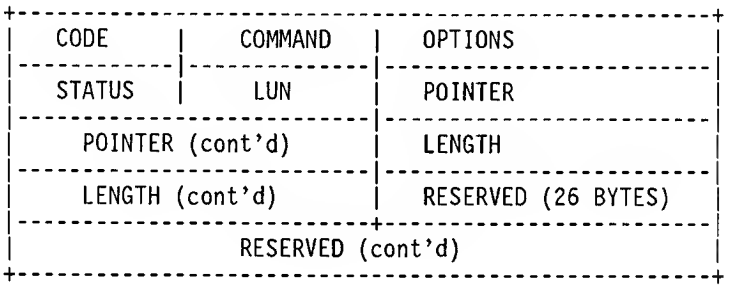
2.2.1.3 Code $01 Commands.

BIT	MEANING
7	Retrieve-Attributes
6	Fetch-Directory-Entry
5	Fetch-Device-Mnemonics
4	Change-LUN-Assignment
3	Fetch-Default-Volume
2	Reserved
1	Reserved
0	Reserved

The options code for the Fetch-Default-Volume (Command = $08) is defined as:

VALUE	MEANING
0	Return system default volume
1	Return temporary default volume
2	Return spooler default volume
3	Return default volume for this session

The parameter block format for the Fetch-Device-Mnemonics call (command = $20) is:



where:

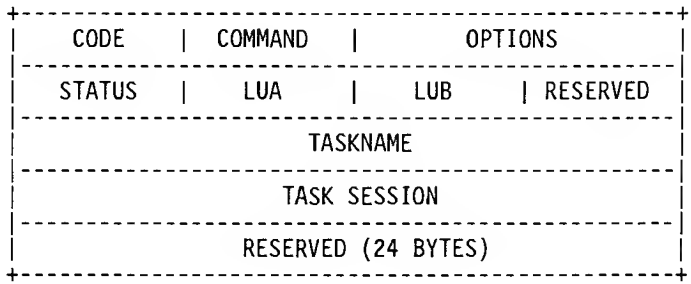
pointer — contains the address of a receiving buffer.

length — is the number of bytes in the buffer. Refer to paragraph 2.3.10 for the buffer format.

2.2.1.4 Code $01 Options. The options code is defined as:

VALUE	MEANING
0-3	Not used
4	Return random access devices
5	Return interactive devices
6	Return printer devices
7-$F	Not used

The parameter block format for a Change-LUN-Assignment call (command = $10) is:



LUA denotes the calling task logical unit number. LUB denotes the called task LUN.

The option code is defined as:

Bit 0	Change LUN direction: 0 = Send LUN 1 = Receive LUN
Bits 1-14	Reserved
Bit 15	Whether existing assignment should remain: 1 = Keep assignment

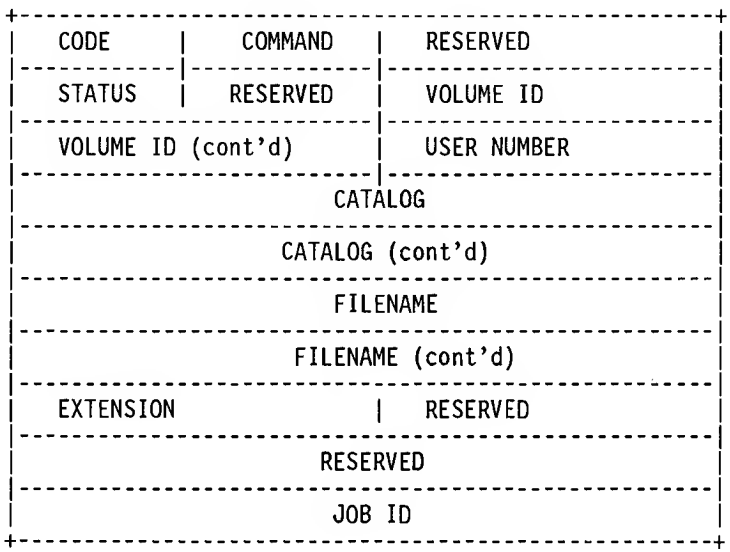
2.2.1.5 Code $02 Commands. Code $02 commands are defined for the spooler task. These commands are invalid if the spooler task is not loaded.

BIT	MEANING	
7	Cancel	Cancel a spooler job
6	Continue	Continue a spooler job
5	Forms	Change forms ID for a spooled device
4	Print	Add a job to the spooler output queue or output a spooler job now
3	Copies	Change number of copies for a spooler job
2	Queue	Display the spooler queue
1-0	Reserved	

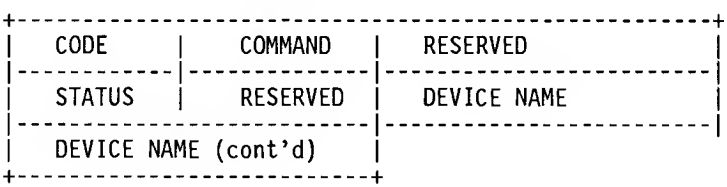
Any reference made to a non-spooler filename from code $02 commands must be the complete file descriptor of voln, user number, catalog, filename, and extension. Any reference made to a spooler filename is the filename and extension. A spooler job ID is a 4-byte field. The first 2 bytes are for the binary user number, and the second 2 bytes is the binary job number.

2

For the Cancel call (command = $80), the format of the parameter block is:



For the Continue call (command = $40), the format of the parameter block is:



For the Forms ID call (command = $20), the format of the parameter block is:

CODE	COMMAND	RESERVED
STATUS	RESERVED	DEVICE NAME
DEVICE NAME (cont'd)		RESERVED
RESERVED		
RESERVED		
RESERVED		
RESERVED		
RESERVED		
RESERVED		
FORMS ID		

For the Print call (command = $10) and the Copies call (command = $08), the parameter block is:

CODE	COMMAND	BINARY OF COPIES
STATUS	RESERVED	VOLUME ID
VOLUME ID (cont'd)		USER NUMBER
CATALOG		
CATALOG (cont'd)		
FILENAME		
FILENAME (cont'd)		
EXTENSION	RESERVED	
RESERVED		
JOB ID		
DEVICE NAME		
FORMS ID		

Used by Print command to specify a non-spooler filename not in spooler queue.

For the Queue call (command = $04), the parameter block is:

CODE	COMMAND	OPTIONS
STATUS	RESERVED	DEVICE NAME
DEVICE NAME (cont'd)		USER NUMBER
USERS CONSOLE NAME		

2.2.1.6 Code $80 Commands. All code $80 commands are privileged and require a system task to execute them.

BIT	MEANING
7-1	Reserved
0	Assign-Default-Volume

2.2.2 Error Status

The status byte interpretation depends on the command specified in the call and is defined under the description. A 0 status means the desired options were performed without error. Table 2-1 gives a summary of all possible error codes. The error status is also returned in register D0. Certain device dependent disk I/O errors return the PSN in error in the size field. The Z bit of the condition code is set or cleared to indicate no error or an error return, respectively. The format of the error status in D0 is:

BITS	VALUE
31-27	3
26-8	0
7-0	Error code (same as status field) ($180000XX)

The first error detected causes the call to return. If multiple functions were specified (e.g., Allocate and Assign), some functions may have been performed properly (always in left-to-right sequence).

TABLE 2-1. FHS Return Status Codes

ERROR CODES	MEANING
$00	No error
$01	Operating system task does not exist
$02	Invalid command
$03	Invalid logical unit
$04	Volume error
$05	Duplicate filename
$06	File descriptor error
$07	Protect code error
$08	Record length error
$09	Shared segment error
$0A	Insufficient directory space
$0B	Access permission error
$0C	Insufficient system space
$0D	Invalid assignment
$0E	Invalid device type
$0F	Invalid transfer method
$10	Invalid taskname
$11	Invalid buffer address
$12	Invalid file type
$13	Internal error
$14	Invalid parameter block address
$15	Data block error
$16	Size error
$17	Non-existent filename
$18	End of directory
$19	Key error
$1A	FAB length error
$1B	Default volume not defined
$1C	File not ready to output
$1D	User number not owner or user #0
$80-FF	I/O errors

2.2.3 Logical Unit Number (LUN)

This byte defines the LUN used for all the FHS functions except Allocate, Delete, and Fetch-Default-Volume.

2.2.4 Volume ID

This 4-byte ASCII field identifies the volume for direct-access devices or the device mnemonic for non-direct-access devices. The Allocate, Assign, Delete, and Retrieve-Attributes functions require this field.

The volume ID field, the user number, catalog, filename, and extension fields identify a file descriptor. The system returns either the volume ID or device name in the volume ID field on a Retrieve-Attributes call. Volume ID is a required field with default assignments (blank fill). Refer to paragraph 1.3.1 for a description of default handling. If the default volume ID is requested in an Allocate, Assign, or Delete request, the actual value is returned in the parameter block. A Fetch-Default-Volume request returns the appropriate default volume in the volume ID field.

When assigning a device (such as CN10 or PR) to a LUN, the device name appears in the volume ID field, left-justified and blank filled on the right. The catalog, filename, and extension fields are ASCII blanks and the user number is 0.

2.2.5 User Number

This 2-byte field is provided for manipulating data created under the user environment. It is a method of ownership. The system returns the user number on a Retrieve-Attributes call; it is 0 for a non-direct-access device. Default values are described in paragraph 1.3.1. If the default user number is requested on an Allocate, Assign, or Delete request, the actual value is returned in the parameter block.

2.2.6 Catalog Name

Catalog Name is an 8-byte ASCII field and is part of the file identification on a direct-access device; it is not required for a non-direct-access device. The catalog name is left-justified and filled with blanks to the right. The user must specify the catalog name for Allocate, Assign, Rename, and Delete calls. The system returns the catalog name on a Retrieve-Attributes call; it is blank for a non-direct-access device.

2.2.7 Filename

Filename is an 8-byte ASCII field that is part of the file identification on a direct-access device; it is not required for a non-direct-access device. Filename is left-justified and filled with blanks to the right. The user must specify the filename for Allocate, Assign, Rename, and Delete calls. The system returns the filename on a Retrieve-Attributes call; it is blank for a non-direct-access device.

2.2.8 Extension

This 2-byte ASCII field typically identifies the file type (e.g., RO, LO, SA, etc.) on a direct-access device. It is treated as the filename field and it is required under the same conditions. Refer to paragraph 1.3.

2.2.9 Two-Byte Reserved Field

This field must be initialized to 0 and is reserved for future enhancement.

2.2.10 Write/Read-Protect Code

Access protection codes for direct-access files and devices are specified using this word. The Allocate, Assign, Protect, and Delete functions require these codes. The Retrieve-Attributes call uses this field to return the device or file attributes.

2.2.11 Record Length

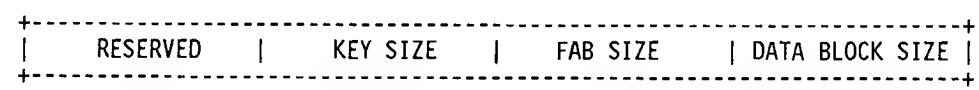
This word field, on an Allocate call, must contain the logical record length for a non-contiguous file. If 0, the file has variable length records; otherwise, it must be an even number from 2 to 65,280 ($FF00).

On a Retrieve-Attributes or Assign call, the file's logical record length or device's physical record length is returned in this field. Record length is not used for other functions.

2.2.12 Size/Pointer

The size field is defined for the Allocate call, depending on the type of file being allocated. For a contiguous file, the size field must be a longword containing the file size in sectors.

For a noncontiguous file, the size field is redefined to four 1-byte fields:



The key size is only used for indexed sequential files. It specifies the length of the key field and must be an even number between 4 and 100 inclusive. For indexed sequential files with duplicate keys, the range is 0 through 100.

The FAB size contains the size in sectors of each file access block. The FAB size specified at allocate must be a number between 0 and 20 inclusive. If the FAB size is specified as 0, File Management Services (FMS) allocates one-sector FABs.

The data block size contains the size in sectors of each data block in the file. The data block size specified at allocate must be a number between 4 and 255 (inclusive) or 0. If the data block size is specified as 0, the FMS allocates four sector data block sizes.

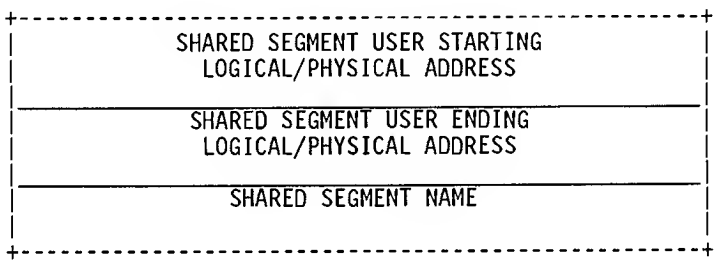
On a retrieve attributes or assign call for a file, the size field is returned in the format described above. For direct access devices, size returns the number of sectors on the device. Size is not used for a non-direct-access device.

2

On a Fetch-Directory-Entry request, the size field is treated as a pointer to a 60-byte receiving buffer whose format is described in paragraph 2.3.10.

2.2.13 Shared Segment Information

If the shared segment option (bit 5) is requested on an Assign call, the parameter block is expanded at the end by 12 bytes. The format of this information is:



Unless bit 11 is set, the starting logical address must be passed by the caller. When bit 11 is set, FHS returns the starting logical address of the segment to the user, which is the same as the physical address. Regardless of the setting of bit 11, FHS returns the segment name and the logical ending address. With the shared segment option set, FHS allocates a segment to be used for the data block and allows the caller to access the segment by attaching the user to the segment. When FHS attaches the segment to the user, the user's starting logical address is used if bit 11 is 0. If bit 11 is set, FHS attaches the user to the segment, forcing the logical address of the segment to be the same as the physical address. When the file is closed, FHS detaches the caller. The segment size is determined by the data block size specified at allocation. The user must request a shared segment if performing block writes to a noncontiguous file, or if record and block I/O is mixed. The shared segment option is ignored if the assignment is to a contiguous file or volume.

2.3 FHS CALL DESCRIPTIONS

2.3.1 Allocate

The Allocate function reserves space on a direct-access device and in the directory for the specified file type. For a contiguous file, the entire file extent is reserved. For noncontiguous files, only a directory entry is reserved. The access protect codes are entered in the directory. The required parameters are:

Volume ID: Specifies the volume on which the file is to be allocated. It must be the name of an online direct-access volume; otherwise, "volume error" is returned.

User number: Assigns each filename a number from 0 to 65,533 with established ownership by system or user ID.

Catalog/filename/ extension: Gives the newly allocated file its name. No other file can be on the specified volume under the specified user number with this name; otherwise, "duplicate filename error" is returned.

Write/read protect code: Sets up the initial protection codes for the file. Defaults to unprotected.

Record length: Use this field when allocating a noncontiguous file. Sets the logical record length for the file; any even number up to 65,280 bytes. This number must be less than or equal to the data block size for the file and once established it cannot be changed. Specification of a 0 record length indicates the file contains variable length records (refer to paragraph 1.6.2). A user other than user 0 cannot allocate files on a volume owned by someone else. Any user can allocate files on the system volume (if its volume user number is 0).

Size: Contiguous file:

This size is the entire file in sectors. It may be any size up to the maximum amount of contiguous space available on that volume at that time. If the requested size is 0, "size error" is returned. If more space is requested than exists on the volume, "insufficient space" is returned.

2

Noncontiguous file:

This is the data block size for the file in sectors. It can be any size up to the maximum set at initialization time for the system, but never greater than 255. If this parameter is too large, it may be difficult to open the file. It also includes the size of the FAB. The key size is included for indexed sequential files.

File type: Contiguous, sequential, indexed sequential with duplicate keys, or indexed sequential without duplicate keys.

A file may be allocated either from the terminal or from a user program via FHS. If either the default volume or user number is requested, the actual assigned values for both are returned to the caller.

Applicable error codes are:

$02	Invalid command; attempt to allocate spooler file by non-system task.
$04	Volume error; the specified volume was not mounted.
$05	Duplicate filename.
$06	File descriptor error; syntax error.
$07	Protect error; user number conflict.
$08	Record length error; Record length is not an even number or record length too large for data block size.
$0A	Insufficient directory space.
$0B	Permission error; entire disk currently assigned exclusive read/write.
$0E	Device type error; the specified volume is not a direct-access device.
$12	Invalid file type.
$14	Invalid parameter block address; outside user space.
$15	Data block length error; data block length less than 4.
$19	Key length error; key size less than 4 (0 is acceptable if duplicate keys) or greater than 100 or not even or greater than record length.
$1A	FAB length error; FAB length greater than 20.
$80-FF	I/O error.

2.3.2 Assign

2.3.2.1 Assign Function. The Assign FHS call from a user program or the Assign command from a terminal assigns a file or device to a logical unit. At the time of assignment, the system allocates (out of system space) a File Control Block (FCB) and buffers for noncontiguous files. The buffer space required, if any, depends on the file's data block size and FAB size. If the file's block size for the remaining system space is too great, the file may not be opened. Here, it returns an "insufficient system space" error status. When a space error occurs, the user program can close another open file assignment, freeing some system space and allowing the first file assignment to be retried.

Files should not be kept open longer than necessary and unless there are other overriding considerations, the file's physical block lengths should be kept as short as possible.

The Assign function establishes a logical connection between a file (or device) and the task through a specified logical unit under a given access permission. The call proceeds as follows.

First, examine the access permission to determine which access protect codes to check. The proper protect code is then checked against the codes in the file directory. For a read assignment, neither the owner of a file nor user 0 need match the read-protect code, all other users must. Only the owner or user 0 can issue a write assignment. The write-protect code must match. An assignment with a write-protect code of $FF is allowed to support file rename and protection changes. If there is no conflicting assignment, the file is then assigned according to the requested access permission. For all volume and file assignments except sequential files, any write access permission is changed to EREW. For sequential files, multiple write accesses are allowed for appending to the end of the file. If a write assignment is specified for a noncontiguous file and the overwrite option was selected, all file space is deallocated when the first write request is made. If a sequential file is assigned with multiple write assignments and more than one overwrite request, the first overwrite is processed and the others are ignored. If the option to position at end is selected, the current record pointer is set to the logical end-of-file so that records can be appended by doing Write NEXT requests.

If the shared segment option is requested, FHS allocates a data buffer and attaches the calling task to it. If the assignment is for a contiguous file or volume, the shared segment option is ignored. The required parameters are:

Access permission
LUN
Protect codes (Wcode and/or Rcode, depending on access permission)
Volume ID
User number
Catalog name
Filename
Extension

User number, catalog name, filename, and extension fields are not used for non-direct-access devices, and should be initialized with spaces (except user number field that should be set to 0).

2

For an assignment to a device marked as a spooler device, and spooling is initiated, a forms ID may be specified. The forms ID field is redefined as the first 4 bytes of the catalog field. If this field is spaces, standard forms ID is used by the spooler. Otherwise, the 4-byte content is used as the forms ID for the spooler.

The information returned to the caller for an assignment to a file or volume is:

Record length
Size
File type
Device type (Bits 10-8 of option word)
User attributes (Bits 15-12 of option word)

In addition, the volume ID and user number are returned if the default was requested for either. For a temporary or spooler file assignment, the system generated filename is returned to the caller. These fields are returned in the FHS parameter block in the positions required by the Allocate command.

Applicable error codes are:

$03	LUN error; invalid LUN.
$04	Volume error; no such volume.
$06	File descriptor error; syntax error.
$07	Protect error; mismatch on access protection codes or nonowner attempted write assignment.
$09	Shared segment error; user passed starting logical address conflicts with existing logical addresses or user's segment space full or device assignment with shared segment option selected.
$0B	Permission error; requested permission may not be granted for file, or entire disk is currently assigned exclusive read/write.
$0C	Insufficient system space error; no room for FCB or shared segment.
$0D	Assignment error; LUN already assigned or offline.
$17	Nonexistent filename.
$80-FF	I/O error; interpreted as IOS error codes defined in Chapter 3.

2.3.2.2 Temporary File Creation. The system recognizes temporary files. These files are temporary because they are deleted when closed.

A temporary file can be created by using FHS:

If an Allocate or Assign call specifies a filename with a first character of ampersand (&), the FMS generates a temporary filename. If volume identification is not specified, the file is placed on the temporary volume. If a temporary volume ID has not been specified via operator command, the file is placed on the default system volume. Using & for a first character and appending an internal counter creates a unique filename and extension. This filename is placed in the filename field of the FHS parameter block.

An attempt to allocate only or to assign only automatically allocates a temporary file and assigns it to the specified LUN. Required parameters are:

User number
File type
LUN
Record length
Size
& as first character of the filename

2.3.3 Change-Access-Permission

This function allows the user to change the current access permission of an assigned file or device. The LUN and access permission portions of the options field are required. In addition, if the access permission is being changed from non-read to read, the read-protect code must be provided. The write-protect code must be provided if going from non-write to write.

The requested new access permission cannot, however, conflict with existing assignments. For example, if the file was assigned for EW, an access permission requiring write access is not allowed. Refer to Table 1-1 for information about compatible access permissions.

If an error is encountered while processing this request, the file remains assigned with its original access permission.

Applicable error codes are:

$03	LUN error; invalid LUN
$07	Protect error; user number/protect code conflict
$0B	Permission error; new permission cannot be granted
$0D	Assignment error; LUN not assigned

2.3.4 Rename

The Rename function changes an assigned filename or device name. For a file rename, the file must be assigned for EREW. The required parameters are:

LUN
User number
Catalog name
Filename
Extension

If the Rename call is successful, the specified user number, catalog name, filename, and extension fields replace the previous user number, catalog name, filename, and extension fields in the directory. For a device rename, if the device supports both read and write, the assignment must be for EREW. If the device supports only read, the assignment must be for ER. If the device supports only write, the assignment must be EW. For a device rename, the required parameters are LUN and the new device ID. Only system tasks can rename non-direct-access devices.

Applicable error codes are:

$03	LUN error; invalid LUN
$05	Duplicate filename
$06	File descriptor error
$07	Protect error; file protected by protect codes
$0B	Permission error; file not assigned for EREW
$0D	Assignment error; LUN not assigned
$80-FF	I/O error as returned by IOS

2.3.5 Protect

This function changes an assigned file's access protection codes. The required parameters are protect codes and LUN. The given LUN must be assigned to a direct-access file. The file must be assigned for EREW. If a device is specified that supports both read and write, the device must be assigned for EREW. If the device only supports read, the assignment must be ER. If the device only supports write, the assignment must be for EW (unless the caller is a privileged task that may modify the protection codes of non-direct-access devices). If the call is rejected, the previous protect codes remain unchanged.

Applicable error codes are:

$03	LUN error; invalid LUN
$0B	Permission error; not assigned for EREW
$0D	Assignment error; LUN not assigned

2.3.6 Close

This function discontinues an assigned logical connection between a task and a file or device. LUN is the only required parameter. The file's directory entry is updated and the specified LUN is unassigned. A busy status is returned for logical units with outstanding I/O to haltable devices. The system waits for any incomplete write data transfers to terminate and writes out any partly filled buffers to the volume. After a Close, temporary files are automatically deleted. The allocated data segment for the file is deallocated. If it was a shared segment (specified at assignment), FHS detaches the calling task from it.

Applicable error codes are:

$03	LUN error
$0D	Assignment error; LUN not assigned
$80-FF	I/O error as returned by IOS

2.3.7 Delete

This function deletes the file's directory entry by zeroing out the first character of the filename field, and releases the space on the disk previously occupied by the file. For a Delete function, the volume ID, user number, catalog name, filename, and extension and protect code parameters must specify a direct-access file that is not currently assigned. Only the owner or a privileged task can delete a file, provided the write-protect code is matched and the file is not delete-protected ($FF). If these conditions are met, the file is deleted from the directory of its volume. The sectors previously occupied by the file on the disk are marked as available in the sector allocation map.

If either the default volume or user number was requested, the actual assigned values for both are returned to the caller.

Applicable error codes are:

$04	Volume error; no such volume
$06	File descriptor error
$07	Protect error; invalid protection codes
$0B	Permission error; file not closed, or entire disk currently assigned exclusive read/write
$0E	Device type error; non-direct-access device
$17	Non-existent filename
$80-FF	I/O error as returned by IOS

2.3.8 Checkpoint

The Checkpoint call performs the buffer-clearing and directory-updating functions of a Close call without performing other functions of a Close operation. This is a protective operation to guard against system failure for either critical files or files running for lengthy time periods without being closed.

The Checkpoint function empties the buffered FMS buffers by writing to the file (or by copying the buffers to the user's input buffer) and updates the directory entry for an indexed file. LUN is the only required parameter. Requesting a Checkpoint for a contiguous file or for a non-direct-access device has the same effect as an IOS Wait-Only call (refer to paragraph 3.3.3).

The applicable error codes are:

$03	LUN error
$0D	Assignment error; LUN not assigned
$80-FF	I/O error as returned by IOS

NOTE

The user may use checkpointing after sensitive data is added to a buffered file. Logical blocking of data in memory in system buffers leaves the file vulnerable. Checkpointing can preserve the data's integrity on the direct-access device. In case of system failure, all data on files up to the latest Close or Checkpoint operation is recoverable; data appended after the most recent Checkpoint is lost. Checkpoint differs from a Close/Assign sequence because repositioning is not performed. Filename, access permission, and protect codes need not be specified.

2.3.9 Retrieve-Attributes

For proper operation, certain programs may require knowledge of the device or file physical attributes associated with a given LUN; for example, to know if random access is possible or if the device is rewindable. The Retrieve-Attributes function gives the user access to this information.

Applicable error codes are:

$03	LUN error; invalid LUN
$0D	Assignment error; LUN not assigned

Some fields within the FHS parameter block are redefined for this call. The only required parameter is the logical unit. The system returns information in fields:

Options
Volume ID or Device Name
User number
Filename
Extension
Size
Protect codes
Record length

The write/read code word is redefined to receive an attributes word (refer to Table 2-2). Any bit set means the device or file supports the corresponding attribute.

TABLE 2-2. FHS Device Attributes Word

BIT	MEANING
15-11	Reserved
10	Supports spooling
9	Printer device
8	Interactive device
7	Supports filemark
6	Supports position record
5	Supports halt I/O
4	Supports image
3	Supports random
2	Supports binary
1	Supports write
0	Supports read

Bits 1 and 0 reflect the current access permission. For example, if a file is assigned with read only access permission, it has bit 1 reset in the device attributes.

If the record length is fixed, the system sets the record length field to the physical record length associated with the device (e.g., 80 or 132 for a line printer, 256 for floppy/hard disk); these 2 bytes are set to 0 for a variable record length device (e.g., magnetic tape). For the contiguous file, the record length is set to 256. For a noncontiguous file, it is the logical record length chosen for that file at its allocation time.

The volume ID, user number, catalog, filename and/or extension for a named file, or the device mnemonic for a device are returned in the file descriptor portion of the parameter block.

The current size of a contiguous file is returned in the size field. The size field is formatted as described for the Allocate function (refer to paragraph 2.3.1) for noncontiguous files. The number of physical sectors is returned for a disk or volume assignment. For a non-direct-access device, size contains the page length (i.e., the number of lines per page). For a device with no page length, size is set to 0. These sizes are returned as unsigned binary numbers.

The options code most significant byte is set to indicate the file or device type. Table 2-3 lists the codes for all supported devices. In addition, for files, the user attributes are returned in the options code in the position required by the Allocate function.

TABLE 2-3. Device Codes

CODE	FILE OR DEVICE TYPE
0	Contiguous file
1	Sequential file
2	Indexed sequential file (no duplicate keys)
3	Indexed sequential file (duplicate keys allowed)
20	Serial port on VERSAmodule 01
21	Parallel port on VERSAmodule 01
30	Interactive terminal on IPC interface (Motorola EXORterm 155)
31	Non-EXORterm 155 terminal on IPC
35	Interactive terminal on local driver (Motorola EXORterm 155)
36	Non-EXORterm 155 terminal on local driver
60	Magnetic tape
90	Low speed line printer on IPC
91	High speed line printer on IPC
95	Low speed line printer on local driver
100	Asynchronous communications line
255	Null device

To obtain information about a device, do a Configuration-Status request (TRAP #2) which returns a channel type code field and a device type code field in the configuration status block. The device type code field displays the type of device; terminal, printer, tape, floppy disk, or hard disk. The attributes word and parameters provide more detailed information about the device configuration. Refer to paragraph 3.3.9 for more details.

2.3.10 Fetch-Directory-Entry

For a Fetch-Directory-Entry call, the given LUN must be an assignment (usually public read) to a volume ID, i.e., the Assign call must have specified only the volume ID; the rest of the file descriptor must have been initialized to blanks.

The Fetch-Directory-Entry call causes the size/pointer field to be interpreted as a pointer to a 60-byte buffer.

The first/next directory entry is returned in the buffer for each Fetch-Directory-Entry command. The search criteria for family names (refer to paragraph 1.3.2) can be given in the filename fields of the parameter block.

A user number field value of -1 defaults to the logon user number. A user number of -2 defaults to all user numbers only for the Fetch-Directory-Entry call.

An asterisk (*) in any position of the catalog name, filename, or extension field means match any character.

The format of the 60-byte block returned to the user is:

0	USER NUMBER	
2	CATALOG	
10	FILENAME	
18	EXTENSION	
20	RESERVED	
22 24	FILE START	
26 28	FILE END	
30 32	LOGICAL SECTOR END OF FILE	
34 36	LOGICAL RECORD END OF FILE	
38	WRITE CODE	39 READ CODE
40	ATTRIBUTES	41 LAST DATA BLOCK SIZE
42	LOGICAL RECORD LENGTH	
44	RESERVED	45 KEY SIZE
46	FAB SIZE	47 DATA BLOCK SIZE
48	DATE FILE ALLOCATED	
50	DATE FILE LAST ASSIGNED	
52 58	RESERVED	

The format of the attributes field is:

Bits 7-4 User attributes

Bits 2-0 File type

3 = Indexed sequential (with duplicate keys)
2 = Indexed sequential (without duplicate keys)
1 = Sequential
0 = Contiguous

Applicable error codes are:

$0B Access permission error; LUN not assigned to volume.

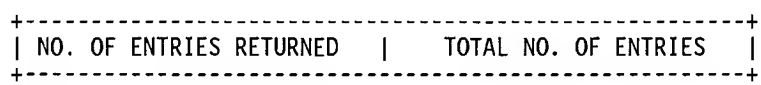
$11 Invalid data buffer; address in size field not in user segment space or not in read/write segment.

$18 End of directory; no more directory entries exist.

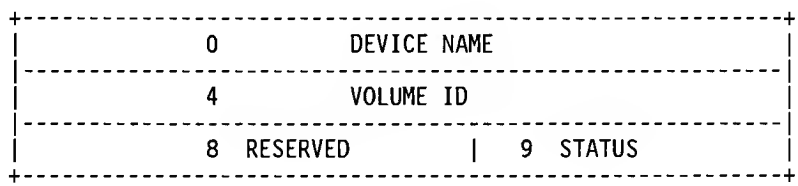
$80-FF I/O error as returned by IOS.

2.3.11 Fetch-Device-Mnemonics

A logical unit assignment is not required for the Fetch-Device-Mnemonics call and can be executed at any time. The FHS block pointer field points to a buffer whose length is contained in the size field. Device entries are returned in the buffer. The size field is modified on return to indicate the number of entries returned in the buffer, as well as the total number of entries that exist for the request. The format of the size field on return is:



The format of each entry returned is:



The volume ID is returned if the device is random access and if a volume is currently mounted. If the volume ID is 0, there is no volume ID associated with the device.

The status byte is defined:

BIT	MEANING
7-3	Reserved
2	Device status changed
1	Write-protected
0	Offline

This call is used to find out the existence of devices known to the system. After the device name is found, it can be assigned and a Retrieve-Attributes call placed to obtain its functionality. The call would be helpful in being able to send messages to all interactive devices. For a random access device, the user should do the assignment by specifying the volume ID unless that field is 0. In that case, the device name should be used.

Applicable error codes are:

$0F Buffer overflow; user buffer too small to contain all requested device mnemonics.

$11 Invalid data buffer; user's return buffer not in his space or address is odd.

2.3.12 Change-LUN-Assignment

The Change-LUN-Assignment call is an intertask function and requires coordination between two tasks. (The Change-LUN-Assignment parameter block is described in paragraph 2.2.1) Two functions are provided:

Send request

LUA contains the LUN for the existing assignment. LUB contains the LUN for the new assignment and must not be assigned. The taskname/session contains the ID of the task that will receive the new assignment. If the taskname/session is all zeros, the ID of the task that called FHS is used (i.e., a task is changing its own LUN assignment). The sending task must be a system task if the sessions are not the same. As a result of this operation, the calling task LUN is closed and the called task LUN is now assigned. Only a system task can send LUN 0.

Receive request

LUA contains the LUN to receive the assignment and must be not assigned. LUB contains the LUN of the existing assignment. Only a system task can receive a LUN 0. The taskname/session contains the ID of the task that owned LUB. As a result of this operation, LUB is closed. The called task must be DORMANT, PAUSED, or SUSPENDED. Note that a task can close another task's LUN by issuing an FHS Receive-LUN and an FHS Close.

With the keep option set, the send request is converted to a send and keep and the receive to a receive and keep. In neither case is the original LUN assignment closed.

Applicable error codes are:

$02 Invalid function.

$03 LUN error.

$07 Protect code error; on a change and keep request, the assignment was for a temporary file.

$09 Shared segment error; attempt to change LUN for file with shared segment.

$0B Access permission error; on a keep request, the original LUN assignment was for exclusive access.

$0C Insufficient system space; on a keep request for a file, insufficient buffer space exists.

$0D Assignment error.

$10 Invalid taskname.

2.3.13 Fetch-Default-Volume

The Fetch-Default-Volume call does not require a logical unit assignment, and can be executed at any time. On return, the volume ID field is modified to contain the requested default volume.

Applicable error codes are:

$02 Invalid function.
$03 Default volume not defined.

2.3.14 Cancel

The Cancel function is defined for the spooler task. This function deletes the associated queue entry for the spooler task, stops output if currently being output, and deletes the file if it is a spooler file. A user other than user 0 cannot cancel queue entries owned by someone else. User number must be specified as either a numerical user number or two ASCII stars '**'. If user 0 specifies two stars, all files in the queue will be deleted, regardless of user number. If any other user specifies two stars, an error will be generated. If no other fields are specified, all entries for that user are cancelled. To cancel one job, specify either filename or job ID. All unused fields are to be set to binary 0.

Applicable error codes are:

$02	Spooler task not loaded.
$13	File being output now but unable to send an event to the task doing the output.
$17	Non-existent filename; non-existent job ID.
$1D	User number not owner or user #0.
$80-FF	I/O error as returned by IOS.

2.3.15 Continue

The Continue function is defined for the spooler task. It is used to start output of a spooler file to a spooler device after a Forms ID command or after an I/O error was encountered and corrected while outputting. Device name must be specified.

Applicable error codes are:

$02	Spooler task not loaded.
$0E	Invalid device name specified.
$13	Not able to send an event to the task doing the output, on pending I/O error.
$80-FF	I/O error as returned by IOS.

2.3.16 Forms

The Forms function is defined for the spooler task, and is used to change the forms ID of a spooler device. After the current job being output to the spooler device is completed, nothing is output until a Continue command is issued. At that time, only jobs with matching forms ID are output. Both device name and forms ID parameters are required. Forms ID is a 4-byte ASCII field.

Applicable error codes are:

$02	Spooler task not loaded.
$0E	Invalid device name or forms ID specified.
$80-FF	I/O error as returned by IOS.

2.3.17 Print

The Print function is defined for the spooler task. It has two purposes:

a. To add a non-spooler filename to the spooler queue for output.

b. To start output of a filename already in the spooler queue, overriding the file's existing output priority.

The user number parameter is required. A user other than user 0 cannot print queue entries owned by someone else. To start output of a filename already in the spooler queue, specify filename or job ID parameters. To add a non-spooler filename to the spooler queue for output, the filename descriptor and device name parameters are required. The forms ID parameter contains the forms ID wanted, or binary 0 for the default forms ID of STND. The number-of-copies field contains the binary number of copies to output. If this parameter is binary 0, and a non-spooler filename is being added to the spooler queue, the default number of copies is 1. If the filename is already in the spooler queue and the number-of-copies parameter is binary 0, the existing number of copies are output; otherwise, the parameter value is used.

Applicable error codes are:

$02	Spooler task not loaded.
$17	Non-existent filename.
$1C	File not ready to output.
$1D	User number not owner or user #0.
$80-FF	I/O error as returned by IOS.

2

2.3.18 Copies

The Copies function is defined for the spooler task. This function is used to change the number of copies for a spooler queue entry to output. The user number and number-of-copies parameters are required. Job ID or filename parameters must also be specified. The number-of-copies parameter is the binary number of copies to output. If this parameter is binary 0, one copy is output. A user other than user 0 cannot change the number of copies for a queue entry owned by someone else.

Applicable error codes are:

$02	Spooler task not loaded.
$17	Non-existent filename.
$1D	User number not owner or user #0.
$80-FF	I/O error as returned by IOS.

2.3.19 Queue

The Queue function is defined for the spooler task. It is used to view the contents of the spooler queue and the status of any spooler devices that have had assignments made to them. Any user may view the spooler queue. The applicable parameters are option, device name, user number, and user's console name. User's console name is required. To view the entire queue, set option, device name, and user number parameters to binary 0's. To view queue entries of the current session, set the option parameter to binary 1 and the device name and user number parameters to binary 0's. To view the queue entries for a specified device, set the device name parameter to a spooler device name and the option and user number parameters to binary 0's. To view queue entries for a specified user, set the user number parameter to the binary user number wanted and the option and device name parameters to binary 0's. To view the spooler devices with their current forms ID setting and current status, set the device name parameter to FORM and the option and user number parameters to binary zeros.

Applicable error codes are:

$02	Spooler task not loaded.
$80-FF	I/O error as returned by IOS.

CHAPTER 3

IOS - INPUT/OUTPUT SERVICES

3.1 GENERAL

A task calls Input/Output Services (IOS) to do all general-purpose I/O. The information describing the required activity is passed to the IOS module in a standard IOCB parameter block. When a task requires a change to the existing configuration of peripheral attributes and parameters before execution of the I/O, the necessary information is first passed to the IOS module in a separate configure IOCB parameter block (refer to paragraphs 3.4.1 and 3.4.2).

IOS calls are:

a. Read from a file or device.

b. Write to a file or device.

c. Write to then read from a device.

d. Update a record.

e. Delete a record.

f. Format a disk/sector.

g. Position to a particular record of a file.

h. Rewind or position to the beginning of a file.

i. Terminate outstanding I/O.

j. Place a task into I/O WAIT.

k. Request an attention event for a break condition.

l. Send a break to LUN specified.

m. Suspend a calling task until the specified device is free.

n. Proceed with concurrent task execution.

o. Release a task from break services.

p. Set device-specific configuration.

q. Change device-specific parameters assigned at SYSGEN.

r. Return device configuration.

3.2 THE STANDARD IOCB

The parameter block shown in Figure 3-1 is used for requesting I/O with a peripheral when the existing configuration of parameters and attributes for that peripheral does not have to be changed for the I/O.

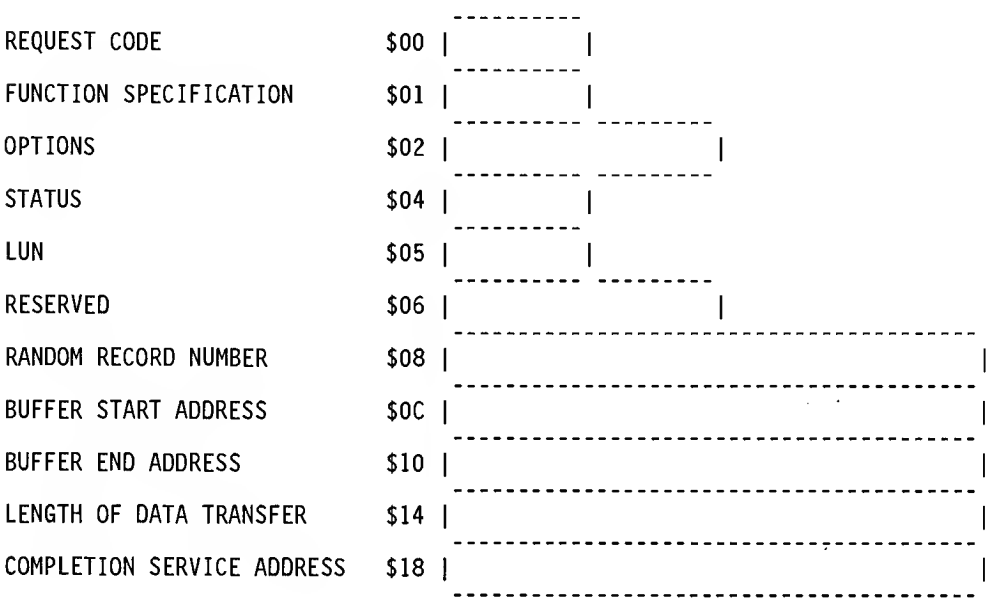


FIGURE 3-1. The Standard IOCB

Because the error status is returned to the parameter block, a standard IOCB parameter block must be on a word boundary and in the writable segment of the program address space. The following example shows how to code an IOS call using this parameter.

```
IOSBLK:  DC.B    CODE     Request code
         DC.B    FCT      Function specification
         DC.W    OPT      Options
         DC.B    0        Status
         DC.B    LUN      Logical unit
         DC.W    0        Reserved
         DC.L    RECNUM   Random record number
         DC.L    START    Start address
         DC.L    END      End address
         DC.L    0        Length of data transfer
         DC.L    RETURN   Completion address
```

All fields are not required for every request. For some requests, a field may be redefined. Refer to paragraph 3.2 for individual field details. The parameter block address is passed to IOS in the A0 register.

For each IOS function, Appendix B specifies the required parameter fields for each request code. The entire parameter block must reside within one Memory Management Unit (MMU) segment.

3.2.1 Request Code Field

The Request Code Field specifies data transfer requests, command requests, and privileged requests. Table 3-1 defines this field.

TABLE 3-1. Request Code Field Values

VALUE	MEANING
$00	Data transfer requests
$01-$02	Command requests
$04-$3F	Not used
$40	Nonvalidated command
$41-$7F	Not used
$80	Privileged requests
$81-$FF	Not used

If the request code byte does not indicate a valid request, the system immediately returns an invalid function code ($02) to the user. By using the Retrieve-Attributes FHS call, the user can determine which commands the logical units support.

3.2.1.1 Data Transfer Functions (Request Code $00). Table 3-2 defines the function specification field for each data transfer request. The bit value must be 1 to start the corresponding function.

TABLE 3-2. IOS Data Transfer Functions (Code $00)

BIT	MEANING
7	Reserved
6	Transmit-Break
5	Format
4	Delete-Record
3	Update-Record
2	Output-with-Input
1	Write
0	Read

If an invalid bit setting is specified, the request is rejected as an illegal function. Refer to status field description (refer to paragraph 3.2.2). For example, a function specification field value of $01 specifies a request for read.

3.2.1.2 Command Functions Request (Request Codes $01 and $02). Table 3-3 defines the function specification field for command requests, code $01. Table 3-4 defines the function specification field for command requests, code $02.

TABLE 3-3. IOS Command Functions (Code $01)

BIT	MEANING
7	Configure-Device
6	Configuration-Status
5	Local-Break-Claimer (Break Service)
4	Halt-I/O
3	Wait-Only
2	Test-I/O-Complete
1	Rewind
0	Position

Claim/Negate Driver Events:

These level 2 command requests allow a task to receive events directly from an I/O device driver and to cancel this facility. The exact structure of the event is driver-dependent. Not all device drivers support every command.

TABLE 3-4. IOS Command Functions (Code $02)

BIT	MEANING
7	Reserved
.	.
.	.
.	.
2	Negate driver events
1	Claim driver events
0	Negate-Local-Break-Claimer (Break Service)

Chapter 4 lists the specific commands supported by individual I/O devices under the description for each device. Paragraph 1.8 explains the implementation of command functions for direct-access files.

Level 40 Request Codes:

The purpose of the level 40 function is to allow an I/O driver to handle all function validity tests. The IOS level 40 function handler has no knowledge of the command, options, or parameter block structure. It does handle basic IOS checks -- LUN validation, parameter block length, and reserved field zero. Assignment must be to a driver only, not a file.

3.2.1.3 Privileged Requests. For a request code $80, a supervisor task can communicate with the IOS module for privileged access. This call's function specification and options fields will be redefined. Request code $80 is not executable by a user task. Table 3-5 defines the function specification field for Privileged Requests.

TABLE 3-5. IOS Command Functions (Code $80)

BIT	MEANING
7	Reserved
.	.
.	.
.	.
1	Configure-Defaults (System Function - Unclaimed Breaks)
0	Reserved

3.2.1.4 Options. The Options Field specifies the type of transfer to be performed. Table 3-6 defines the option code field for each data transfer request.

TABLE 3-6. IOS Data Transfer Options

BIT	MEANING
15	TASK ID
	0 No task ID supplied.
	1 Task ID supplied.
14-13	LOGICAL ACCESS/POSITION
	00 Access/position to next record.
	01 Access/position to current record.
	10 Access/position to prior record.
	11 Access/position to random record specified in Random Record Number (RRN) field.

TABLE 3-6. IOS Data Transfer Options (cont'd)

BIT	MEANING
12	FORMAT OPTION
	A Used with Format command to disk.
	0 Format entire disk. 1 Format track (requires Physical Sector Number (PSN)).
	B Used with Read or Output-with-Input commands to terminal.
	0 Do not flush type ahead buffer. 1 Flush type ahead buffer before doing read.
	C Used with Write command to terminal.
	0 Do not clear the "discard output" condition. 1 Clear the "discard output" condition before doing read.
	D Used with Configure-Device command
	0 Do not dismount the device. 1 Dismount the device.
11	RETRY/FORMAT OPTIONS
	A Used with Read/Write commands to disk.
	0 Retry the command. 1 No retries.
	B Used with Format command to disk.
	0 Do not format the track as bad.
	1 Format the track as bad (requires PSN of the first bad sector in the track). (No other option bit may be set.)
	C Used with Configure-Device command to disk.
	0 Use error correction. 1 Do not use error correction.
10	INPUT FORMATTED/IMAGE (see NOTE)
	(Valid only for the "input" action of Output-with-Input commands)
	0 Format input according to device and state of bit 0. 1 Do not format (image mode).

TABLE 3-6. IOS Data Transfer Options (cont'd)

BIT	MEANING
9	COMPLETION/SERVICE ADDRESS
	0 Return task normally.
	1 Return task to the address specified in the completion field.
8	RETURN KEY WITH RECORD
	0 Do not return key to the user on a record Read request. Ignore unless indexed sequential file type.
	1 Return key to the user on a record Read request.
7	LOGICAL RECORD/RANDOM KEY ACCESS
	0 Perform logical record I/O by logical record number.
	1 Perform random record I/O by key value. Valid only for indexed sequential files. Ignore if bit 6 is set.
6	RECORD/BLOCK ACCESS
	0 Perform record access.
	1 Perform block access.
5	SUPPRESS ECHO
	0 Do not suppress echo.
	1 Suppress echo.
4	BREAK NOTIFICATION
	(Valid only for I/O termination)
	0 Task receives shared notification.
	1 Task receives exclusive notification.
3	OUTPUT FORMATTED/IMAGE
	(Valid for Read and Write requests and the output action of the Output-with-Input request)
	0 Format request according to the device/file and the setting of bit 0. Ignore if contiguous file or volume assignment.
	1 Do not format request (image mode). In image mode, the user must include all carriage returns and line feed operations explicitly.

TABLE 3-6. IOS Data Transfer Options (cont'd)

BIT	MEANING
2	CONNECTION WAIT
	0 Place task in Connection Wait state until the requested device/file is free. At that time, process request.
	1 Reject request if the requested device/file is not free.
1	WAIT/PROCEED
	(Indicates the action taken after the I/O has been initiated)
	0 Wait. Put task in I/O Wait until the data transfer is complete.
	1 Proceed. Return control to task after initiation of I/O.
0	CONFIGURE/FORMAT OPTIONS
	A Used with Configure-Device command
	0 Mount device. 1 Configure device, but do not mount.
	B ASCII/BINARY options (Indicates the type of formatting requested)
	0 ASCII formatting requested. 1 Binary formatting requested.
	If bit 3 is set or file data type is not ASCII, this bit is ignored.

NOTE: The term "formatted" (as opposed to "image") refers to a data transfer between program and peripheral. Formatted or image mode data transfer can be specified for Write, Read, and Output-with-Input.

Details of formatted and image modes depend on the particular device. In general, formatted mode is more convenient and people-oriented. For example, formatted input from a terminal includes several useful line-editing features. Image mode provides a more transparent data transfer, i.e., fewer interpretations and restrictions are placed on the data. This makes the data transfer less convenient for the human user but far more flexible if the user is talking to a device rather than to a person. In this mode, if some format is desired on input or output, the caller must supply all EOL markers or other codes the peripheral requires, and accommodate any format requirements. For example, a value of $16003 in the function and options field specifies a request for: read, binary formatting, proceed, random access.

3.2.2 Status Field

The system returns the execution status of the requested function in the status field byte. If an error occurred, this status byte is set to indicate the general type. If an error did not occur, it is set to 0. The status code is also returned in register D0. Certain device dependent disk errors return a sector number in the Random Record Number (RRN) field. The Z bit of the condition code register is set if no error occurred; it is cleared if there was an error.

The status code format returned in D0 is shown in Table 3-7.

TABLE 3-7. Register D0 Status Format

BIT	VALUE
31-27	2
26-8	0
7-0	Status byte as returned in IOS parameter block ($1000000XX)

Chapter 4 explains the specific interpretations of the status codes as applied to devices. Table 3-8 gives the general definition of the status byte values.

TABLE 3-8. Interpretation of IOS Status Byte Value

CODE	MEANING
$00	No error
	*** Syntax Errors ***
$81	Operating system task does not exist
$82	Invalid function
$83	Invalid logical unit
$84	Invalid data buffer address
$85	Invalid random record
$86	Invalid parameter block address
$87	Protect code error
$88	Configuration parameter block error
	*** Device-Independent Errors ***
$C1	Buffer overflow

TABLE 3-8. Interpretation of IOS Status Byte Value (cont'd)

CODE	MEANING
$C2	End of file
$C3	End of volume
$C4	Invalid FAB
$C5	Invalid transfer for device
$C6	Break condition
$C7	Internal error
$C8	FAB/data block conflict
$C9	Record does not exist
$CA	Record already exists
$CB	Record overflow/too many records in data block
$CC	Key error, FAB/key conflict
$CD	Insufficient disk space
$CE	Unrecoverable file error
$CF	File space allocation/deallocation conflict
	*** Device-Dependent Errors ***
$D1	Unrecoverable device error
$D2	Data compare error (sector number returned)
$D3	Sector protect error (sector number returned)
$D4	Device not mounted
$D5	Beginning of tape detected (magnetic tape only)
$D6	Input formatter error (General Purpose Interface Bus (GPIB) only)
$D7	Tape already mounted (magnetic tape only)
$D8	TMS9914A handshake error (GPIB only)
$D9	Invalid command/mode combination (GPIB only)
$DA	Invalid command for busable device (GPIB only)
$DB	Run-time error (GPIB only)
$DC	Invalid sector address (GPIB only)
$DD	Invalid request for pending interrupt (GPIB only)
$DE	Invalid command for bus (GPIB only)
$DF	Data transfer aborted (GPIB only)
$E1	Device not ready
$E2	Device/file busy
$E3	Data CRC error (sector number returned)
$E4	Write-protected device
$E5	Deleted data mark detected (sector number returned)
$E6	Timeout
$E7	Invalid sector address
$E8	Checksum error
$E9	Disk restore error
$EA	Data overrun
$EB	Device status changed
$EC	Track/section ID not found (sector number returned)
$ED	Address mark CRC error (sector number returned)
$EE	Seek error
$EF	Bad sector (sector number returned)

TABLE 3-8. Interpretation of IOS Status Byte Value (cont'd)

CODE	MEANING
	*** Channel Errors ***
$F1	Channel busy
$F2	Channel DMA error
$F3	Unrecoverable channel error
$F4	Controller error
$F5	Device configuration
$F6	DMA bus error
$F7	DMA mapping error
$F8	DMA controller error
$F9	Indeterminate channel error

Generally, for a data request, if "invalid function", "device unavailable", or "invalid LUN status" is indicated alone, then data was not transferred. "Device unavailable" may indicate that the device is physically inoperative. If the device becomes unavailable after a transfer is started, the error is set to indicate that some data may have been transferred.

3.2.3 Logical Unit Number (LUN)

To provide device-independent I/O, all I/O requests are directed to a LUN. LUN is a number from 0 to 255 that describes an entry in the task's LUN table. The maximum LUN for a system is determined at system generation. Before executing the IOS call, the operator command or FHS call must assign the particular device or file to the specified LUN. The call is rejected if an invalid or unassigned LUN is specified. If no operation is desired, the specified LUN should be assigned to the null device. An I/O operation to the null device is effectively a no-op, and is useful during program development.

3.2.4 Two-Byte Reserved Field

This 2-byte field is initialized to zero and is reserved for future enhancements.

3.2.5 Random Record Number (RRN)

The RRN field specifies the logical record number (starting at 0) to be accessed on data transfer requests if the random record access option is set. The interpretation of this field is device-dependent and function-dependent.

If the RRN field is minus one on a random logical record position request to a file, the current record pointer is positioned to the last record in the file. The current record pointer is returned in the RRN field only for file position requests.

If the function byte indicates Output-with-Input, this field is redefined to the start buffer address for the input function (refer to paragraph 3.3.8 "Output-with-Input").

3.2.6 Buffer Address

For data transfer requests, the buffer-start and buffer-end fields specify the buffer. The buffer-start address is the first byte in the buffer and the buffer-end address is the last byte in the buffer.

A buffer must start on a word boundary and must be fully contained in a user segment of the task address space. A buffer used in Read requests must be in a writable segment because the Read operation changes the memory locations. For random key access, the buffer contains the requested key.

An IOS invalid address error occurs if:

Buffer-start and buffer-end addresses are not in the user logical segment.

Buffer-end address is less than buffer-start address.

Buffer for a Read request is in a write-protected logical segment.

3.2.7 Length of Data Transfer

The length-of-data-transfer field returns the actual number of bytes transferred during a request, either Read or Write. This field is most useful when dealing with variable length records. The field contents are undefined if an error status is indicated, except for "buffer overflow" where the transfer length does reflect the number of characters transferred for a Read request. On a Position request, this field contains the offset to the start of the record in the data block.

For an Output-with-Input request, this field serves a dual purpose: the initial value is the user-supplied input buffer length; on I/O completion, that value is overlaid with the length of the input data actually received (as described above).

When reading data from a Multi-Channel Communications Module (MCCM) serial port, if the number of bytes transferred is odd, a space is added to the end of a data transfer.

3.2.8 Completion/Address

The completion-address field specifies a return address for I/O completion if the completion address option was selected. When an I/O request is initiated (with Proceed), setting bit 9 returns control to the task at the completion address.

If the request is Proceed-I/O and the calling task is set up to receive an event, the Asynchronous Service Queue (ASQ) automatically returns control to the task at the address specified (refer to Chapter 2 in the M68000 Family Real-Time Multitasking Software User's Manual).

If the request is Proceed-I/O and the calling task is not set up for task queue entry (no ASQ has been set up), the completion address is ignored.

3.3 IOS CALL DESCRIPTIONS

3.3.1 Connection-Wait-I/O

Use a Connection-Wait-I/O request when a task is waiting for the requested operation. The system coordinates I/O requests so that only one I/O request may access any device or file at a time. If Connection-Wait is requested and the specified device/file is in use at the time of the data transfer request, the IOS system suspends the calling task until the device is free and initiates the I/O request. If Connection-Wait is not specified and the device is in use, the IOS request is rejected and the error status is set to device busy. The calling task may retry the request at a later time. If the specified device is not in use, setting the Connection-Wait option does not affect the request.

3.3.2 Proceed/Wait-I/O

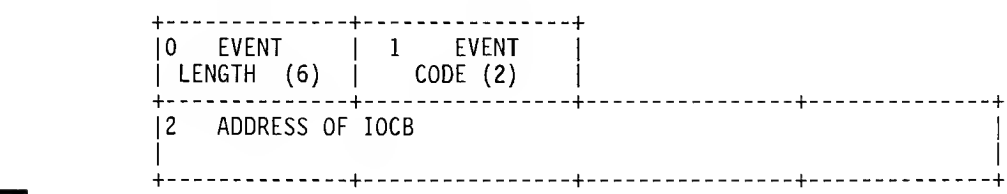
When an I/O request has been initiated (that is, the specified device is free), any Proceed-I/O call returns control to the task so that the task may concurrently execute with the data transfer. Except for invalid function and invalid LUN, which are rejected before transfer initiation, the status is not set until I/O completion. Check for invalid function and invalid LUN status before proceeding with the concurrent task execution.

Check the status of the I/O request by performing one of the following:

a. Monitor (poll) the status field in the parameter block.
b. Accept an event in the I/O task's ASQ on completion.
c. Issue a Wait-Only request to the same LUN to wait for completion.

An Wait-I/O call requests the IOS to suspend the calling task until completion of the I/O operation.

The format for an I/O completion event sent to the calling task is:



3

3.3.3 Wait-Only

A Wait-Only request function places the task into I/O Wait until the completion of a previous Proceed-I/O request to the specified LUN. If a task does not have any outstanding I/O to the specified LUN, control is immediately returned. This call uses the function, LUN, and status fields of the IOS parameter block. Invalid LUN is the only error status this call returns. The status of the I/O being tested is returned in the parameter block associated with the original Proceed-I/O call, not in the Wait-Only call's parameter block.

3.3.4 Halt-I/O

A Halt-I/O request attempts to cancel an Proceed-I/O request previously issued. This is useful on an interactive terminal device. If the Halt-I/O request is not used, any outstanding I/O request must be satisfied before any other I/O can be started on a device.

The printer and CRT devices support the Halt-I/O request.

When issuing a Halt-I/O request, the IOS schedules the I/O operation for termination. The termination is asynchronous to the Halt-I/O request. When the I/O terminates, the task receives an event in its ASQ, if enabled. The completion event is identical to the I/O completion event described in paragraph 3.2.8. The address of the IOCB contained in the returned event is the address of the original data transfer parameter block and not the address of the Halt-I/O parameter block.

There are two unique IOS parameter blocks involved in the Halt-I/O processing:

a. The data transfer parameter block.

b. The Halt-I/O command function parameter block.

The user specifies the data transfer block when the I/O is initiated and the Halt-I/O parameter block when the halt is requested.

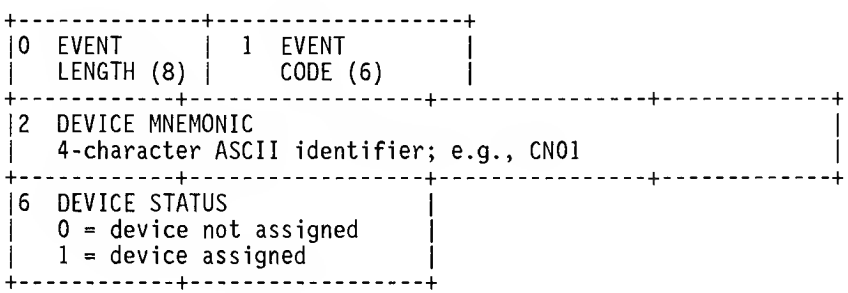
Note that the condition codes returned are unpredictable when a Halt-I/O is requested. Instead, the user should monitor the status of the Halt-I/O command function parameter block. The pseudo-code that follows represents a routine that issues a Halt-I/O request and does not return until the I/O is halted or it is recognized that there is no I/O to halt. The condition codes resulting from the routine show whether the I/O is halted or not.

```
Pre-set the status byte in the Halt-I/O parameter block with $FF.
Issue the TRAP #2 call to halt the I/O.

WHILE

     Status byte of Halt-I/O parameter block equal $FF.
     WAIT (use RELINQUISH directive)

ENDW

IF

     Status byte of Halt-I/O parameter block is 0 then the I/O was halted.

     {In addition to the above there will be an $E6 in the status byte of
      the data transfer parameter block and a I/O termination event in the
      requestor's ASQ, if enabled}.

     Return to caller.

ELSE

     There was no I/O to halt.

     {There will be an $82 in the status byte of the Halt-I/O parameter
      block.
      An I/O completion event in the requestor's ASQ, if enabled.
      The status byte of the data transfer parameter block will reflect
      the status of the I/O completion.}
     Return

ENDIF
```

3.3.5 Local-Break-Claimer

This request applies only to interactive devices. When a break condition is present on the device specified, an attention event is sent to the requesting task (Local-Break-Claimer).

The format for a break attention event is:



If bit 1 in the device status word is set, the specified device is assigned.

The only option applicable is the option for service address and and it uses the standard IOS parameter block. Required entries are:

Function
Options
LUN
Status

3.3.6 Negate-Local-Break-Claimer

The Negate-Local-Break-Claimer request applies only to interactive devices. A task that previously requested break service (Local-Break-Claimer) may obtain release from break service responsibility via this request. The standard IOS parameter block is used and the required entries are:

Function
LUN
Status

3.3.7 Test-I/O-Complete

A Test-I/O-Complete call returns with a condition code of "Z bit = 1" if there is no outstanding Proceed-I/O to the specified LUN by the task. If an outstanding Proceed-I/O exists, the call returns an error condition code of "Z bit = 0".

3.3.8 Output-With-Input

An Output-with-Input operation on an interactive device that supports this call, allows a task to write to a device and issue a read for response at the same time. When a program issues an Output-with-Input request, the system writes the data in the user's data buffer for the length specified by the start and end addresses and before returning, issues a read for the length specified in the length field. The RRN field is redefined to indicate the start-buffer address of the input function.

If the file is not opened for exclusive read/write access, an invalid function is returned.

3.3.9 Update-Record

The Update-Record request is valid only for an assignment to a noncontiguous file. An Update-Record request must be used to change an existing record in a file. If the record to be updated does not exist, a "record does not exist" error is returned. If a record Write is requested for an existing record, a "record exists" error is returned; the Update-Record command should have been used. For a sequential file, the new record must be the same size as the existing record, otherwise an "invalid data buffer address" error is returned.

For an indexed sequential file, the new record must have the same key as the old record, otherwise "key error" is returned. "Invalid data buffer address" error occurs if the buffer is not large enough to accommodate the key.

3.3.10 Delete-Record

A Delete-Record request is valid only for an assignment to an indexed sequential file. If the requested record does not exist, a "record does not exist" error is returned. If the file is not opened for exclusive read/write access, an "invalid function" is returned.

3.3.11 Format

This request causes a disk or track to be formatted. The Format track option requires a PSN to calculate the appropriate track number.

The options available are:

Proceed
Track
Alternate track
Completion address

The Format track option uses the RRN field of a standard IOS parameter block for the PSN and the buffer-start address for the alternative track PSN. The required entries are:

Function
Options
LUN
Status

The logical unit assignment must be exclusive read/write for a random access device or volume.

3.3.12 Transmit-Break

This request, which applies only to interactive devices, sends a break condition to the LUN specified. The Transmit-Break request uses the standard IOS parameter block and the required entries are:

Function
Options
LUN

The only applicable options are those for service address and I/O with proceed.

Any attempt to create this condition on a device whose assignment is read-only results in an invalid function ($82) error return. The device drivers that do not allow this function also return the same error.

3.4 CONFIGURATION

3.4.1 General Information

Associated with each terminal, printer, and disk drive are two configurations of parameters and attributes: the default configuration and those parameters and attributes currently in effect. At SYSGEN time the user defines the default configuration that establishes the current configuration for each device when the system is booted.

The IOS module provides three requests for obtaining and changing the current configuration values and for changing the default configuration values.

a. Configure-Device
b. Configure-Defaults
c. Configuration/Status

In the IOS module context, the terms parameter and attribute refer to the discrete parcels of information that are conveyed to the device driver so that the driver can produce the desired peripheral activity. Parameters represent conditions or quantities having several values and are passed to the driver in parameter block fields of one or more bytes. For example, the length of the write timeout period requires a quantity in a parameter field.

Attributes, on the other hand, represent conditions or actions that can have only two values. Consequently, attributes are associated with individual bits within a single field -- the attributes word. An example of an attribute is to use or not to use parity checking.

Because an existing parameter or attribute of a peripheral may or may not need to be changed, a mechanism is provided to inform the device driver whether to take action or not according to the parameter field value or attribute bit state passed in the parameter block. The activating mechanism for changing parameters is the word contained in the parameters mask field in which individual bits correspond to fields further on in the parameter block. If an individual bit value is 1, the driver reads and acts according to the value contained in the corresponding parameter field. If the value of an individual bit is 0, the driver writes the current or default value of the parameter in the corresponding field of the user's parameter block. Figure 3-2 portrays the change parameter mechanism.

NOTE

Not all serial device drivers support all configuration attributes and parameters. Performing a Configuration-Status command returns the set of attributes and parameters supported for that device and drivers.

TERMINAL CONFIGURATION BLOCK

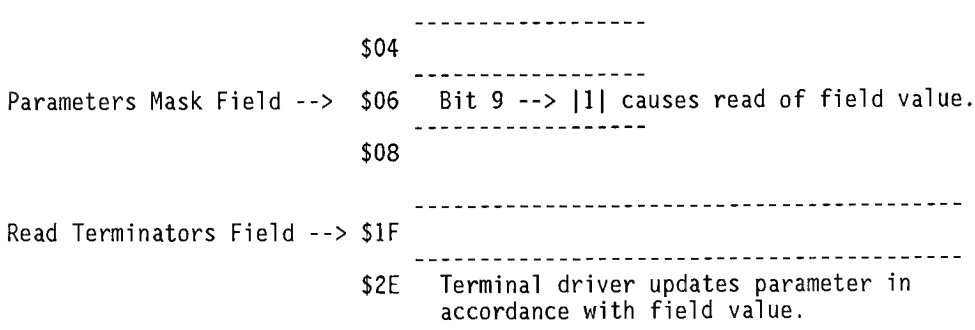


FIGURE 3-2. Change Parameter Mechanism Example

Similarly, the activating mechanism for changing attributes is the attributes mask field. Individual bits within this word length field correspond to individual bits within the attributes word field. If an attributes mask field bit is 1, the driver takes action based on the state of the corresponding bit in the attributes word. If the mask bit is 0, the corresponding bit in the attributes word is given the current or default value of the attribute. Figure 3-3 shows the change attributes mechanism.

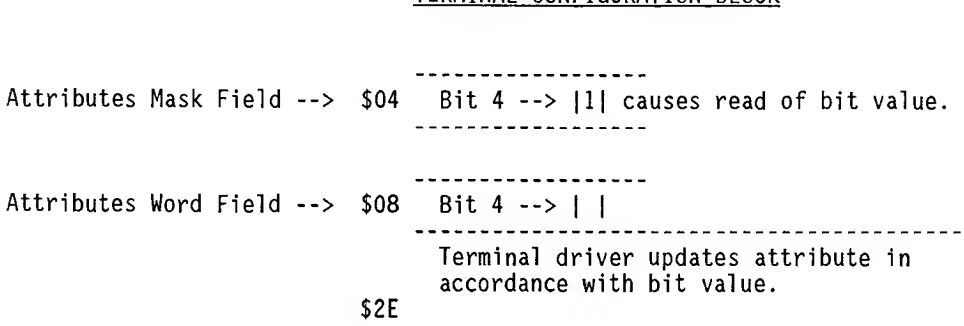


FIGURE 3-3. Change Attribute Mechanism Example

3.4.2 The General IOCB For Configuration Requests

The parameter block shown in Figure 3-4 is used for reviewing or changing the default or current configuration of the peripheral's parameters and attributes before requesting I/O with that peripheral. The Configuration/Status Block (CSB) address field in this parameter block points to the configuration parameter block for a Configure-Device request and for a Configure-Defaults request. For a Configuration-Status request, the CSB address field points to the address where the current configuration values will be returned.

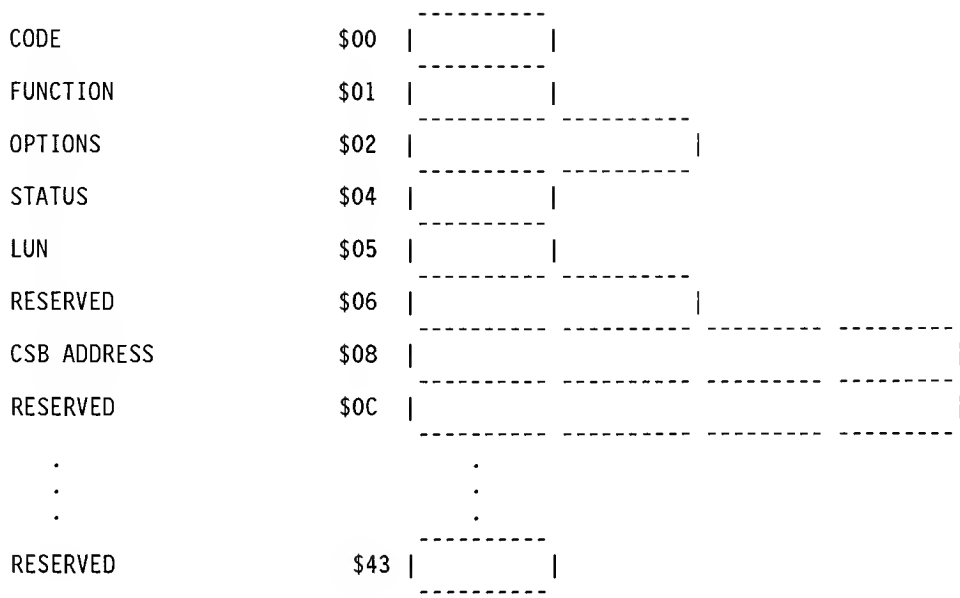


FIGURE 3-4. The General Configuration IOCB

A configure IOCB parameter block must be on a word boundary and in the writable segment of the program address space because the error status is returned to the parameter block. The following example shows how to code an IOS call using the Configure-Device parameter block.

```
IOSBLK:  DC.B    CODE     Request code
         DC.B    FCT      Function specification
         DC.W    OPT      Options
         DC.B    0        Status
         DC.B    LUN      Logical unit
         DC.W    0        Reserved
         DC.L    CSB      CSB address
         DC.L    0        Reserved
         DC.L    0        Reserved
         DC.L    0        Reserved
         DC.L    0        Reserved
```

All fields are not required for every request; the Configure-Device IOCB parameter block address must be passed to IOS in register A0.

3.4.3 The Configure-Device and Configure-Defaults Requests

Any task that has an assignment for the device to be configured can execute the Configure-Device request. The new configuration remains in effect until it is changed by another Configure-Device request. When all assignments for a device whose configuration has been changed have been closed, the configuration reverts to a default configuration previously defined, either at SYSGEN time or by a previous Configure-Defaults request. Assignments to the same device can exist for multiple tasks at the same time. Therefore, one task could issue a Configure-Device request and affect the I/O of other tasks that would be unaware that a change to the configuration had occurred. An attempt to configure a spooler device while spooling is active results in an error.

At SYSGEN time, the user defines the default parameters for all devices in the system. The default configuration is the device configuration in effect when an assignment is made to a device for which no other assignments exist. The user may alter the default configuration for a device at a later time by issuing a Configure-Defaults request from a system task. An attempt to alter a default configuration from a task that is not a system task results in an error ($82). It should also be noted that the write and read timeout values for a device can be altered only via the Configure-Defaults request. An attempt to alter these timeout values using a Configure-Device request results in a configuration error.

If an error occurs in a configuration request, the status returned in the user's IOCB reflects a general code ($88) for a configuration error; however, the user may obtain the detailed error by examining the error code in the status/error field of the configuration parameter block. Configuration errors returned include errors common to all drivers (codes $01-$3F) and errors specific to a particular driver (codes $40-$7F). Codes $80-$FF are reserved. The device type code, channel type code, and driver code fields contain no significant information if an error exists. When a configuration error occurs, the current configuration or default configuration in effect before the error is unchanged. If the configuration request is successful, the status/error, channel type, device type, and driver code fields contain the information associated with each field.

3.4.4 The Configuration-Status Request

The Configuration-Status request may be executed by any task that has an assignment for the device for which the current configuration information is desired. In addition to the returned current configuration values, the status/error, channel type, device type, and driver code fields contain the appropriate information.

The attributes and parameters mask fields returned in the configuration/status parameter block contain the mask of the attributes/parameters that are supported by the driver of the requested device. If a bit is set, the corresponding attribute/parameter is supported. A reset condition indicates that the attribute/parameter is not recognized by the driver. The user, after obtaining the current configuration information, can alter the desired

attributes by issuing a Configure-Device request with appropriate attributes and parameters mask fields. Any attributes/parameters mask bits not supported by the driver must be set to 0 when a Configure-Device or Configure-Defaults request is made; otherwise, a configuration error occurs.

The general IOCB parameter block for a Configuration-Status request is the same as for a Configure-Device request, except that the CSB address field contains an address to which the current configuration values are moved. Specific meanings of bits within the status/error and device type code fields of the configuration parameter blocks returned for terminals, printers, disks, and magnetic tape are shown in paragraphs 3.5.1, 3.5.2, 3.5.3, 3.5.4, 3.5.5, 3.5.6, and 3.5.9. Returned values for all fields in the configuration parameter blocks for terminals, printers and disks are described in paragraph 3.2.1.4. Regardless of the equipment, bit 7 when set means "device not ready".

FIELD	BIT	MEANING
TERMINAL		
Status/error		
On/off Line	7	0 = Indicates terminal is online. 1 = Indicates terminal is offline.
PRINTER		
Status/error		
On/off Line	7	0 = The device is online. This is always the status returned due to restriction of the Centronics interface.
DISK		
Status/error		
Protection	5	0 = The device is not write-protected. 1 = The device is write-protected.
On/off Line	7	0 = The device is online. 1 = The device is offline.
MAGNETIC TAPE		
Status/error		
Tape density	2,1,0	0 = 1600 bpi. 1 = 800 bpi or blank tape. If transport density is not selectable in software, tape density may not be accurate as reported in this status byte. The user must know the tape density if the transport density is controlled from front panel.
Density select	4	0 = Transport density not selectable in software. 1 = Transport density is selectable in software.
Write Protection	5	0 = Device not write-protected. 1 = Device is write-protected.
On/off Line	7	0 = A Configure-and-Mount command has been received for the tape drive. 1 = No Configure-and-Mount command has been received for the tape drive since bootup or last Configure-and-Dismount command.

3.5 CONFIGURATION PARAMETERS

The following tables describe (for the specified device) each field in the parameter block of the Configure-Device, Configure-Defaults, and Configuration-Status requests.

Values for each configuration parameter field are supplied by the user during the SYSGEN process and written into the default configuration table at system initialization.

3.5.1 Terminal Configuration Parameter Block

In general, when a character received by the driver is interpreted as having special meaning according to the current configuration setting, the action associated with that special meaning is performed and the character is discarded. For example, if the XOFF character is received by the driver, it usually is not in the user's read buffer. However, if the character has been defined as a read delimiter, it is placed in the requestor's buffer as data (assuming there is room). The length of the data transfer field in the requestor's I/O parameter block reflects the presence of the read delimiter for image mode read operations but not for formatted mode read operations. Figure 3-5 shows the terminal configuration parameter block.

NOTE

When changing the configuration for a serial port, do not assign any character more than one function. For example, if $13 (CTRL-S) is.configured as the XOFF character and the REPRINT LINE character, one of the two functions cannot work properly.

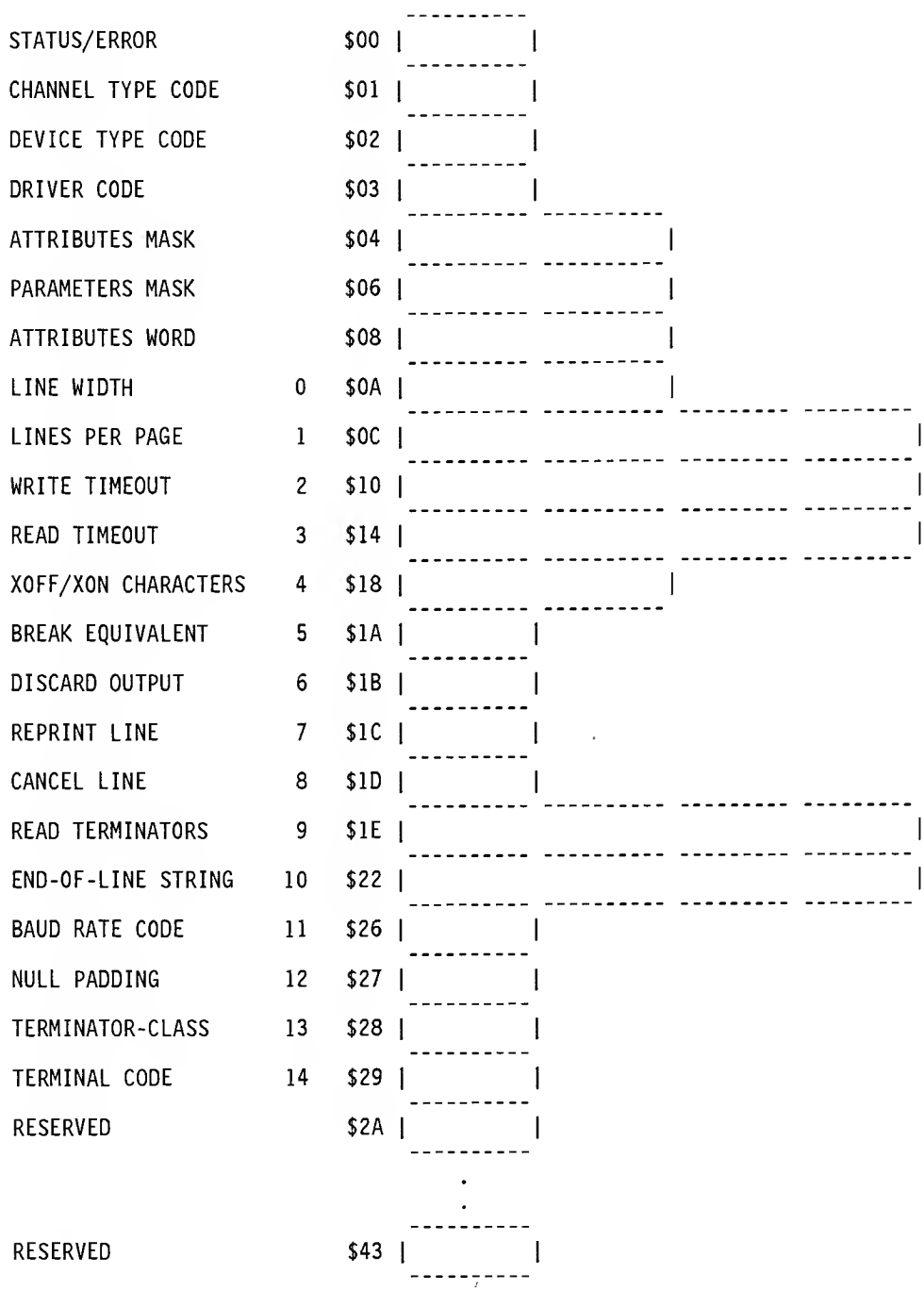


FIGURE 3-5. Configuration IOCB for Terminals

NOTE

The number in the left number column above is the on/off bit in the parameters mask field that corresponds to that parameter.

FIELD	DEFAULT VALUE	IMAGE FORMATTED	(I) (F)	FIELD DESCRIPTION
STATUS/ERROR				Configure-Device Configure-Defaults Configuration-Status On return from a request, this field contains the current device status unless the status field in the user's IOCB is $88 (configuration error), in which case the returned status/error field contains a code that identifies the cause of the error.

IOSCEC Equate	Code	Device Status
CECCDO	$01	Specified parameter/ attribute invalid for Configure-Device request. Use Configure-Defaults request.
CECUAP	$02	Specified parameter/ attribute (Configure-Device/Configure-Defaults request) invalid for driver/ channel type.
CECTTI	$40	Specified parameter/ attribute invalid for Read termination.
CECTIM	$41	Parameter/attribute conflict for modem operation.
CECTBR	$42	Invalid baud rate specified for drive/ channel type or requested baud rate not configurable for driver/channel type.

FIELD	DEFAULT VALUE	IMAGE FORMATTED	(I) (F)	FIELD DESCRIPTION
CHANNEL TYPE CODE				Configure-Device Configure-Defaults Configuration-Status This field contains on return from a request the code for the channel type. If an error is returned in the status/error field this field is reserved for future use. Examples of channel type codes are: $64 7201 A-side on MVME400 $65 7201 B-side on MVME400

FIELD	DEFAULT VALUE	IMAGE FORMATTED (I) (F)	FIELD DESCRIPTION
DEVICE TYPE CODE			Configure-Device Configure-Defaults Configuration-Status If status is returned in the status/error field, this field contains a code identifying the type of device. If an error is returned in the status/error field, this field is reserved for future use. Code — Device Type 3 — Terminal
DRIVER CODE			Configure-Device Configure-Defaults Configuration-Status On return from a request, this field contains the code for the driver type. If an error is returned in the status/error field this field is reserved for future use. Code — Driver 1 — Terminal driver for 7201 ports 8 — Terminal driver for SCN2661 ports $A — Terminal driver for SCN2681 ports $B — Terminal driver for MK68901 ports $C — Terminal driver for R68560 ports $D — Terminal driver for ACIA ports $E — Terminal driver for Mostek 68564 ports $11 — Terminal driver for keyboard/screen on VME/10 $20 — IPC driver

FIELD	DEFAULT VALUE	IMAGE FORMATTED (I) (F)	FIELD DESCRIPTION
ATTRIBUTES MASK			Configure-Device Configure-Defaults A 1 in a particular bit position indicates to use the corresponding bit of the attributes word field to update the configuration. A 0 indicates that the current attribute will be retained. An attempt to configure an unsupported attribute will result in an error. Configuration-Status A 1 in a bit position indicates that the driver supports the corresponding attribute. A 0 indicates that the attribute is not supported by the driver.

Bit	Attribute
0	Hardcopy
1	Use XOFF/XON
2	Bits per character
3	Stop bits
4	Parity usage
5	Type parity
6	Local echo
7	Type ahead
8	Terminate on buffer full
9	Pass nulls
10	Modem/Direct connect
11	Modem off-hook
12-15	Reserved

FIELD	DEFAULT VALUE	IMAGE FORMATTED	(I) (F)	FIELD DESCRIPTION
PARAMETERS MASK				Configure-Device Configure-Defaults A 1 in a particular bit position indicates that the corresponding field from the parameter block should be used to update the configuration. A 0 in a bit position indicates that the parameter value in the current configuration will be retained. Attempting to configure an unsupported attribute results in an error. The following bit/field correspondence is in effect: Bit — Field 0 — LINE WIDTH 1 — LINE PER PAGE 2 — WRITE TIMEOUT 3 — READ TIMEOUT 4 — XOFF/XON CHARACTERS 5 — BREAK EQUIVALENT 6 — DISCARD OUTPUT 7 — REPRINT LINE 8 — CANCEL LINE 9 — READ TERMINATORS 10 — END-OF-LINE STRING 11 — BAUD RATE CODE 12 — NULL PADDING 13 — TERMINATOR-CLASS 14 — TERMINAL TYPE 15 — RESERVED Configuration-Status A 1 in a bit position indicates that the corresponding parameter is supported by the driver. A 0 indicates that the parameter is not supported by the driver.

3

FIELD	DEFAULT VALUE	IMAGE FORMATTED (I) (F)	FIELD DESCRIPTION
ATTRIBUTES WORD			Configure-Device Configure-Defaults Configuration-Status This field contains bit values which determine or reflect the terminal driver's behavior under specific circumstances. The values and their correspondence with terminal attributes are described in paragraph 3.4.2.
LINE WIDTH	80	F	Configure-Device Configure-Defaults When a formatted Write to the terminal is completed, the driver checks to see if it has output a number of printable characters that is a multiple of the line width. If it has, the driver will not output the EOL string as it would normally. The assumption is that the terminal has done its own EOL processing since the last character output was in the last position of a line. Configuration-Status This field contains the current field value.
LINES PER PAGE	24	F	Configure-Device Configure-Defaults The number of printable lines on the terminal screen. This information is not currently used by the driver but is maintained so that a task could retrieve and use it. Configuration-Status This field contains the current field value.

FIELD	DEFAULT VALUE	IMAGE FORMATTED (I) (F)	FIELD DESCRIPTION
WRITE TIMEOUT	900,000	I,F	Configure-Device Illegal. An attempt to change this field using the Configure-Device request results in an error; use the Configure-Defaults request instead. Configure-Defaults The number of milliseconds IOS should allow for completion of a Write request before halting the I/O. The maximum value is equal to 24 hours (86400000). Larger values will not generate an error but will result in a 24-hour timeout value. Configuration-Status This field contains the current field value.
READ TIMEOUT	900,000	I,F	Configure-Device Illegal. An attempt to change this field using the Configure-Device request results in an error; use the Configure-Defaults request instead. Configure-Defaults The number of milliseconds IOS should allow for completion of a Read or Output-with-Input request before halting the I/O. The maximum value is equal to 24 hours (86400000). Larger values will not generate an error but will result in a 24-hour timeout value. Configuration-Status This field contains the current field value.

FIELD	DEFAULT VALUE	IMAGE FORMATTED (I) (F)	FIELD DESCRIPTION
XOFF	$13 (CTRL S)	I,F	Configure-Device Configure-Defaults If the driver received this character, transmission is suspended until it receives the XON character. This is also the character sent by the driver to stop the terminal from transmitting if the XOFF/XON attribute is selected. If this byte is 0, there is no XOFF character. Configuration-Status This field contains the current field value.
XON	$00 (Any Character)	I,F	Configure-Device Configure-Defaults If transmission has been suspended by receipt of an XOFF character, receipt of the XON character resumes transmission. If this byte is 0, then any character resumes transmission. This is also the character sent by the driver to the device to allow it to transmit if the XOFF/XON attribute is selected. Configuration-Status This field contains the current field value.
BREAK EQUIVALENT	(CTRL C) ($03)	I,F	Configure-Device Configure-Defaults Receipt of this character is equivalent to receiving a break condition from the terminal. If this byte is 0 there is no break equivalent character. Configuration-Status This field contains the current field value.

FIELD	DEFAULT VALUE	IMAGE FORMATTED (I) (F)	FIELD DESCRIPTION
DISCARD OUTPUT	$0F (CTRL O)	I,F	Configure-Device Configure-Defaults Receipt of this character causes all writes to the terminal to be discarded until the condition is cleared. The length-of-data-transfer returned for the write is always the entire buffer length, regardless of embedded CR characters. This condition can be cleared by: a. The receipt of another discard-output character from the terminal (a toggle). b. The receipt of any Output-with-Input or Read request from the task. c. The receipt of a Write request with the option set to clear the discard output condition. If this byte is 0, there is no discard output character. Configuration-Status This field contains the current field value.
REPRINT LINE	$04 (CTRL D)	F	Configure-Device Configure-Defaults In response to a Read request, this character redisplays on the next line what has been typed so far. If this byte is 0, there is no reprint line character. Configuration-Status This field contains the current field value.

FIELD	DEFAULT VALUE	IMAGE FORMATTED (I) (F)	FIELD DESCRIPTION
CANCEL LINE	$18 (CTRL X)	F	Configure-Device Configure-Defaults In response to a Read request, this character discards what has been typed so far. If this byte is 0, there is no cancel-line character. Configuration-Status This field contains the current field value.
READ TERMINATORS	$0DDE0000 (CR ..)	I,F	Configure-Device Configure-Defaults The characters specified in this string, when received in response to a Read operation, are interpreted as terminators. A match on any character in this field terminates the Read. The user may specify 0 to 4 characters as terminators. Incoming characters are edited against each character in this field until all characters have been examined or until trailing zeros are encountered. A data string of $00000000 indicates that there are no read terminators. Configuration-Status This field contains the current field value.

FIELD	DEFAULT VALUE	IMAGE FORMATTED (I) (F)	FIELD DESCRIPTION
END-OF-LINE STRING	$0D0A0000 (CR LF)	F	Configure-Device Configure-Defaults The characters specified in this string are echoed when a terminator is received during a formatted Read operation and are sent after each formatted Write operation unless an exact multiple of the line width is reached. The terminator itself is not echoed. Characters are sent until either all have been sent or trailing zeros are encountered. A $00000000 data string indicates no EOL string. Configuration-Status This field contains the current field value.
BAUD RATE CODE	$0E	I,F	Configure-Device Configure-Defaults The codes to indicate the baud rate are: Code — Rate $00 — 50 $01 — 75 $02 — 110 $03 — 134.5 $04 — 150 $05 — 300 $06 — 600 $07 — 1200 $08 — 1800 $09 — 2000 $0A — 2400 $0B — 3600 $0C — 4800 $0D — 7200 $0E — 9600 $0F — 19200 $10 — 38400 $10-FF — Reserved Configuration-Status This field contains the current field value.

FIELD	DEFAULT VALUE	IMAGE FORMATTED (I) (F)	FIELD DESCRIPTION
NULL PADDING	$00	I,F	Configure-Device Configure-Defaults This number, maximum of 255, represents the number of ASCII NUL characters that will be sent following each CR or LF character transmitted by the driver. Configuration-Status This field contains the current field value.
TERMINATOR-CLASS	$00	I,F	Configure-Device Configure-Defaults This field provides the user the ability to terminate Read operations by defining a range of values to act as read terminators. The field is divided into two subfields represented by the high-order 4 bits, the Don't Care Mask, and the low-order 4 bits, the Match Value. The operations described below do not alter the original character being processed. If the Don't Care Mask has a value of 0, it implies that there is no terminator class and the following operations will not be performed. The Don't Care Mask is ANDed with the high-order 4 bits of the incoming character. Then a comparison is made of the ANDed result with the match value subfield. If equal, the character being processed is a terminator character. Configuration-Status This field contains the current field value.

FIELD	DEFAULT VALUE	IMAGE FORMATTED	(I) (F)	FIELD DESCRIPTION
TERMINAL CODE	$00	I,F		Configure-Device Configure-Defaults This field is maintained by the driver but is not used by it. Configuration-Status This field contains the current field value. Code — Terminal Type $00 — EXORterm 155, VME/10 $01-$7F — Reserved $80-$FF — User defined

3.5.2 Terminal Attributes Word Field

A detailed description of the terminal attributes word field follows.

ATTRIBUTE	BIT	DEFAULT VALUE	IMAGE FORMATTED	(I) (F)	BIT DESCRIPTION
Hardcopy	0	0	F		Controls the way a backspace ($08) or delete ($7F) character is echoed in formatted mode. 0 = Back up and erase the character. 1 = Show the deleted characters between angle brackets in a suitable way for hard copy devices. For example, abcd<BS><BS><BS>e would echo abcd<dcb>e. The way the cancel line function is done also changes: 0 = Back up and erase all the characters on the line. 1 = Output a backslash and go down to the next line.

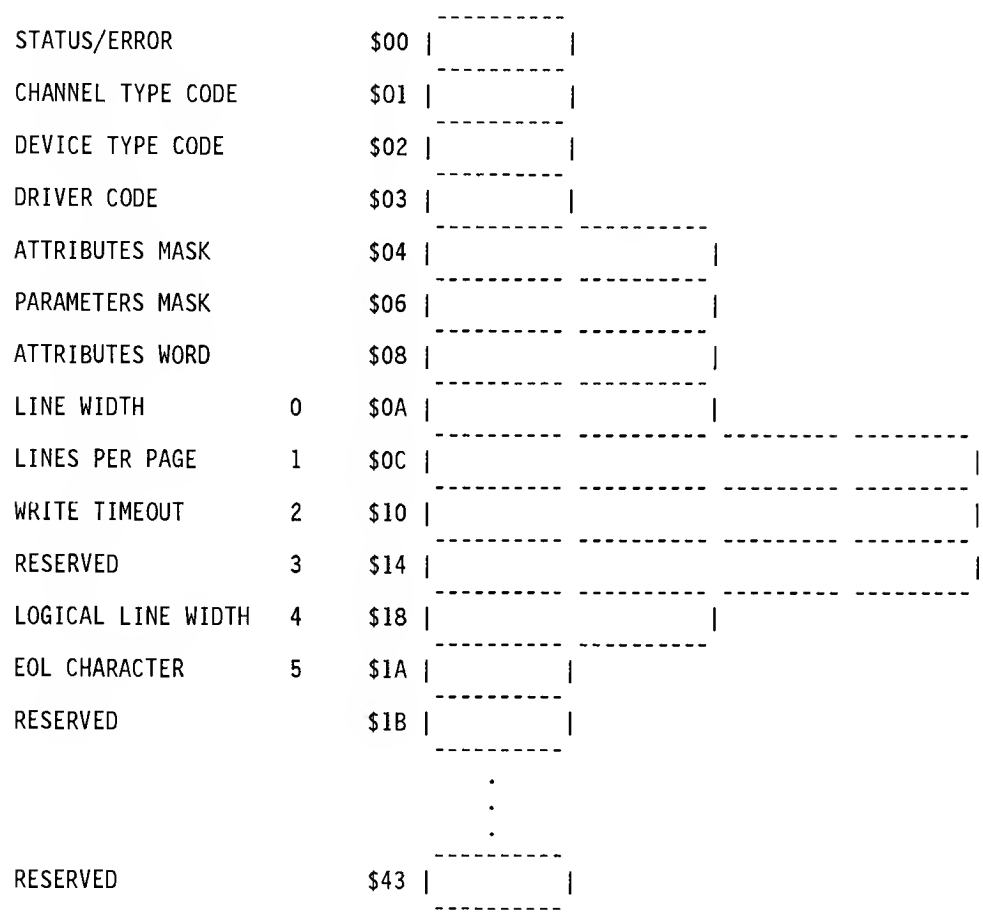
ATTRIBUTE	BIT	DEFAULT VALUE	IMAGE FORMATTED	(I) (F)	BIT DESCRIPTION
XOFF/XON	1	0	I,F		Determines what the driver will use to suspend and resume transmission from the terminal. Modem operation requires that the XOFF/XON bit value = 1. 0 = Use the CTS line. 1 = Use the XOFF/XON characters
Bits/character	2	0	I,F		0 = Transmit and receive 8 data bits per character. 1 = Transmit and receive 7 data bits per character.
Stop bits	3	0	I,F		0 = Follow each character transmitted with one stop bit. 1 = Follow each character transmitted with two stop bits.
Parity	4,5	00	I,F		0 0 = Do not check parity. 0 1 = Do not check parity and drop bit 7. 1 0 = Generate and check odd parity. 1 1 = Generate and check even parity.
Local echo	6	0	I,F		0 = Echo characters received from the terminal, unless the user's read operation has requested that echo be suppressed via an option in the I/O parameter block. 1 = Do not echo characters received from the terminal.
Type ahead	7	0	I,F		0 = Buffer the characters received from the terminal if a read operation is not being processed. If buffer

ATTRIBUTE	BIT	DEFAULT VALUE	IMAGE FORMATTED	(I) (F)	BIT DESCRIPTION
					overflow did not occur, the buffered characters will be processed when the current operation is complete. 1 = Discard the characters received from the terminal if a read operation is not being processed.
Terminate on buffer full	8	0	I,F		0 = Echo the BEL character for each excessive character when the user attempts to overfill the read buffer. 1 = Terminate the read operation when the read buffer becomes full. Characters typed after termination of the read operation will be placed in the type-ahead buffer if the type-ahead option is enabled; otherwise, the characters are discarded.
Pass nulls	9	0	I		0 = During image mode reads, discard ASCII NUL ($00) characters. 1 = During image mode reads, place ASCII NUL characters into the buffer. Formatted mode reads always discard ASCII NUL characters.
Modem/direct connect	10	0	I,F		Used to designate whether the port is connected to a terminal directly or through a modem. If the modem bit value is 1, the XOFF/XON bit value must also be 1. 0 = Port is connected to a terminal. 1 = Port is connected to a modem.

ATTRIBUTE	BIT	DEFAULT VALUE	IMAGE FORMATTED (I) (F)	BIT DESCRIPTION
Modem off hook	11	0	I,F	Used to control the DTR output on the port when connected to a modem. 0 = DTR should be high, allowing modem to answer phone and establish a connection. 1 = DTR should be low, preventing modem from answering phone. If a connection is currently established, modem hangs up phone.
	12-15			Reserved

3.5.3 Printer Configuration Parameter Block

Figure 3-6 shows the Printer Configuration Block.



NOTE: The number in the left number column above is the on/off bit in the parameters mask field that corresponds to that parameter.

FIGURE 3-6. Configuration IOCB for Printers

FIELD	DEFAULT VALUE	IMAGE FORMATTED	(I) (F)	FIELD DESCRIPTION
STATUS/ERROR				Configure-Device Configure-Defaults Configuration-Status On return from a request, this field contains the current device status unless the status field in the user's IOCB is nonzero in which case the returned status/error field contains a code identifying the cause of the error.

IOSCEC Equate	Code	Device Status
CECCDO	$01	Specified parameter/attribute invalid for Configure-Device request. Use Configure-Defaults request.
CECUAP	$02	Specified parameter/attribute (Configure-Device/ Configure-Defaults request) invalid for driver/channel type.
CECPNI	$40	A NUL is an invalid EOL character.
CECPRLZ	$41	Logical and/or physical line width is zero.
CECPLGP	$42	Logical line width exceeds physical line width.

FIELD	DEFAULT VALUE	IMAGE FORMATTED (I) (F)	FIELD DESCRIPTION
CHANNEL TYPE CODE			Configure-Device Configure-Defaults Configuration-Status On return from a request, this field contains the code for the channel type. If an error is returned in the status/error field, this field is reserved for future use.

Code	Type Channel
$11	MCCM IPC printer ports $50-$5F
$50	EXORmacs PIA
$51	VM01 parallel
$52	RTTLIO parallel
$53-$5F	Reserved

FIELD	DEFAULT VALUE	IMAGE FORMATTED (I) (F)	FIELD DESCRIPTION
DEVICE TYPE CODE			Configure-Device Configure-Defaults Configuration-Status If status is returned in the status/error field, this field contains a code identifying the type of device. If an error is returned in the status/error field, this field is reserved for future use.

Code	Device Type
4	Printer

FIELD	DEFAULT VALUE	IMAGE FORMATTED	(I) (F)	FIELD DESCRIPTION
DRIVER CODE				Configure-Device Configure-Defaults Configuration-Status On return from a request, this field contains the code for the driver type. If an error is returned in the status/error field, this field is reserved for future use.
ATTRIBUTES MASK				Configure-Device Configure-Defaults A 1 in a particular bit position indicates that the corresponding bit of the attributes word field be used to update the configuration. A 0 indicates that the current attribute will be retained. Configuration-Status A 1 in a bit position indicates that the driver recognizes the corresponding attribute. A 0 indicates that the attribute is not recognized by the driver.

Code	Driver
2	Printer driver for local PIA port
$20	IPC driver

Bit	Attribute
0	Auto-LF
1	Form feed before print
2	Truncate/wraparound Print at logical line width
3-15	Reserved

FIELD	DEFAULT VALUE	IMAGE FORMATTED (I) (F)	FIELD DESCRIPTION
PARAMETERS MASK			Configure-Device Configure-Defaults A 1 in a particular bit position indicates that the corresponding field from the parameter block should be used to update the configuration. A 0 in a bit position indicates that the parameter value in the current configuration will be retained. The bit/field correspondence in effect is: Bit — Field 0 — Line width 1 — Lines per page 2 — Write timeout 3 — Reserved 4 — Logical line width 5 — End-of-line character 6-16 — Reserved Configuration-Status A 1 in a bit position indicates that the driver recognizes the corresponding parameter. If bit is 0, the driver does not recognize the parameter.
ATTRIBUTES WORD			Configure-Device Configure-Defaults Configuration-Status This field contains bit values which determine or reflect the printer driver's behavior under specific circumstances. The values and their correspondence with printer attributes are described in paragraph 3.5.4.

FIELD	DEFAULT VALUE	IMAGE FORMATTED (I) (F)	FIELD DESCRIPTION
LINE WIDTH	132	F	Configure-Device Configure-Defaults Configuration-Status This value defines the number of printable characters that can be printed on one physical printer line. FORMATTED MODE: In this mode, non-printing characters are passed to the printer but are not counted as part of the physical line width nor as part of the data transfer length. After each printable character is sent to the printer, a test is made to determine if the number of printed characters is a multiple of the physical line width. No action is taken if a multiple has not been printed. If the number is a physical line width multiple and the auto-LF attribute is off, the driver sends an LF character. If the auto-LF attribute is on, the driver assumes the printer will do the appropriate EOL processing. The physical line width may not be 0. IMAGE MODE: In this mode, the line width is ignored. After completion of the request, the data transfer length returned is the length of the data buffer.

FIELD	DEFAULT VALUE	IMAGE FORMATTED	(I) (F)	FIELD DESCRIPTION
LINES PER PAGE	66			Configure-Device Configure-Defaults Configuration-Status This value, number of lines on the printer page, is not currently used by the driver but is maintained so that a task can retrieve and use the information.
WRITE TIMEOUT		I,F		Configure-Device Illegal. An attempt to change this field using the Configure-Device request results in an error. The Configure-Defaults request should be used instead. Configure-Defaults Configuration-Status This field contains the number of milliseconds IOS should allow for completion of a Write request before halting the I/O.

FIELD	DEFAULT VALUE	IMAGE FORMATTED	(I) (F)	FIELD DESCRIPTION
LOGICAL LINE WIDTH	132	F		Configure-Device Configure-Defaults Configuration-Status Logical line width must be less than or equal to the physical line width. FORMATTED MODE: When the logical line width is equal to the physical line width, the driver behaves the same as the physical line width parameter. When the logical line width is less than the physical line width, the driver operates as follows: Non-printing characters are passed to the printer but not counted as part of the logical line width or the data transfer length. After each printable character has been sent to the printer, a test is made to determine if the number of characters printed is an exact multiple of the logical line width. If so, and the auto-LF attribute is also set, an EOL character is sent. If auto-LF is not set, an LF, CR sequence is sent. If the number of characters is not an exact multiple of the logical line width, no LF, CR sequence is sent. Following one of the above actions, printing either resumes or terminates according to the setting of the wraparound/truncate printer attribute.

FIELD	DEFAULT VALUE	IMAGE FORMATTED (I) (F)	FIELD DESCRIPTION
LOGICAL LINE WIDTH	132	F	FORMATTED MODE (cont'd) In general, formatting is provided to match what would be produced if the carriage length of the printer were the same as the logical line width. For example, if using a 132-column printer, a logical line width of 80 provides formatting as though an 80-column printer were being used. IMAGE MODE: In image mode, the logical line width is ignored and the data transfer length returned after completion of the request is the length of the data buffer.
EOL CHARACTER	$0A (LF)	F	Configure-Device Configure-Defaults Configuration-Status FORMATTED MODE: On completion of a write due to an embedded CR in the data buffer or reaching the end of the data buffer, a test is performed to determine if the data transfer length is a multiple of the logical line width. If it is a multiple, no further action occurs. Completed writes whose data transfer length is not a multiple of the logical line width cause the driver to send the EOL character to the printer if the auto-LF attribute has been selected; otherwise, an LF, CR sequence is sent to the printer. The NUL character is not a valid EOL character. IMAGE MODE: The EOL character is not used.

3.5.4 Printer Attributes Word Field

A detailed description of the printer attributes word field is provided below.

ATTRIBUTE	BIT	DEFAULT VALUE	IMAGE FORMATTED (I) (F)	BIT DESCRIPTION
Auto-LF	0	0	F	Used to control EOL processing. 0 = Printer does not support automatic line feed. 1 = Printer supports automatic line feed.
Form feed after assign	1	0	I,F	Used to determine whether a form feed ($0C) is to be sent before the first write after assign. 0 = Form feed is output before first write after assign. 1 = No form feed is sent.
Truncate/ wraparound (print at logical line width)	2	0	F	Used to terminate a print request when the logical line width is reached or to continue printing on the next line. 0 = Continue printing on the next line when the logical line width is reached. 1 = Terminate the print request when the logical line width is reached.

3.5.5 Disk Configuration Parameter Block

The Configure-Device and Configure-Defaults requests for Intelligent Peripheral Controller (IPC) random access devices (VM21) are treated as if a Configuration-Status request had been made. The parameters-mask field and the attributes-mask field must contain zeros. None of the IPC disk configuration parameters may be changed. For non-IPC random access devices, the Configure-Device and Configure-Defaults commands require the parameters-mask field and attributes-mask field to be set correctly. The status command for non-IPC random access devices returns the actual configuration for the device and the media. Figure 3-7 shows the configuration parameter block.

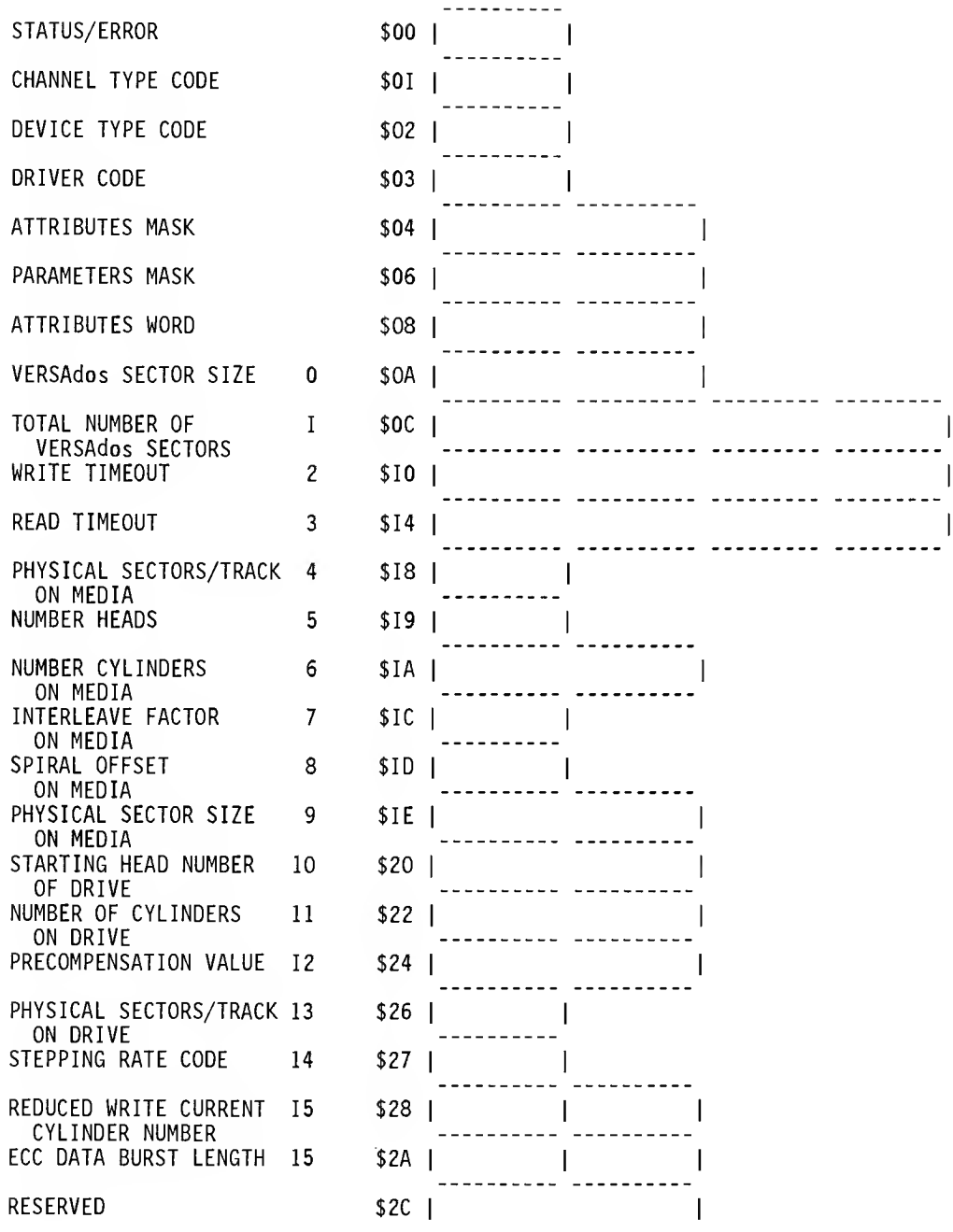


FIGURE 3-7. Configuration IOCB for Disks

NOTE: The number in the left number column above is the bit in the parameter-mask field that corresponds to that parameter.

FIELD	DEFAULT VALUE	FIELD DESCRIPTION
STATUS/ERROR		Configure-Device Configure-Defaults Configuration-Status On return from a request, this field contains the current device status unless the status field in the user's IOCB is nonzero, in which case the returned status/error field contains a code identifying the cause of the error.

IOSCEC Equate	Code	Device Status
CECDAP	$40	Configuration error. Specified parameter not recognized by the drive/channel type.
CECDIA	$41	Specified attribute not recognized by the drive/channel type.
CECDIW	$42	Invalid attributes word specified; conflict between hard disk/floppy disk.
CECDII	$43	Invalid interleave factor specified for the drive/channel type.
CECDSS	$44	Invalid sector size specified for the drive/channel type.
CECDST	$45	Invalid sectors per track specified for the drive/channel type.
CECDDS	$46	Invalid disk size specified for the drive/channel type.
CECDFC	$47	Invalid floppy disk configuration specified for the drive/channel type.

FIELD	DEFAULT VALUE	FIELD DESCRIPTION
CHANNEL TYPE CODE		Configure-Device Configure-Defaults Configuration-Status

On return from a request, this field contains the code for the channel type. If an error is returned in the status/error field this field is reserved for future use.

Code	Type Channel
$10	VM20/VM21 disk IPC
$12	VM22 disk controller
$20	M420 on I/O Channel
$21	RWIN I/O Channel
$23	MVME315 disk controller
$25	MVME320 disk controller

DEVICE TYPE CODE

Configure-Device
Configure-Defaults
Configuration-Status

If status is returned in the status/error field, this field contains a code identifying the type of device. If an error is returned in the status/error field, this field is reserved for future use.

Code	Device Type
1	Floppy disk
2	Rigid disk

DRIVER CODE

Configure-Device
Configure-Defaults
Configuration-Status

On return from a request, this field contains the code for this driver type. If an error is returned in the status/error field, this field is reserved for future use.

Code	Driver
3	M420 disk driver on I/O Channel
4	RWIN disk driver on I/O Channel
6	MVME315 disk driver
9	MVME320 disk driver
$20	VM20/VM21 disk IPC driver
$21	VM22 disk driver

FIELD	DEFAULT VALUE	FIELD DESCRIPTION
ATTRIBUTES MASK	None	Configure-Device Configure-Defaults A 1 in a particular bit position indicates that the corresponding bit of the attributes word field should be used to update the configuration. A 0 indicates that the current attribute is retained. An attempt to configure an unsupported attribute results in an error. The attribute mask bits currently defined are: Bit Field 0 Data density 1 Track density 2 Side/diskette 3 Format type 4 Disk type 5 Disk data density capability 6 Drive track density capability 7 Embedded servo drive seek 8 Post-read/Pre-write precompensation 9 Floppy disk size 10 Alternate sector capability 11-15 Reserved Configuration-Status A 1 in a bit position indicates that the driver supports the corresponding attribute. A 0 indicates that the attribute is not supported by the driver.
PARAMETERS MASK	None	Configure-Device Configure-Defaults A 1 in a particular bit position indicates that the corresponding parameter from the parameter block should be used to update the configuration. A 0 in a bit position indicates that the parameter value in the current configuration will be retained. Attempting to configure an unsupported parameter results in an error.

FIELD	DEFAULT VALUE	FIELD DESCRIPTION
PARAMETERS MASK	None	(cont'd)

The bit/field correspondent in effect is:

Bit	Field
0	VERSAdos sector size
1	VERSAdos total number of sectors
2	Write timeout
3	Read timeout
4	Sectors per track
5	Number of heads on drive
6	Number of cylinders
7	Interleave factor
8	Spiral offset
9	Physical sector size
10	Starting head number on drive
11	Number of cylinders on drive
12	Precompensation cylinder number
13	Sectors per track supported by drive
14	Stepping rate code
15	Reduced write current cylinder number and ECC data burst length

Configuration-Status

A 1 in a bit position indicates that the driver recognizes the corresponding parameter.
A 0 indicates that the parameter is not recognized by the driver.

ATTRIBUTES WORD — None

Configure-Device
Configure-Defaults

This field contains bit values defining all the attributes of the media (even those not recognized by the driver). The attributes word is used with the attributes mask to update the configuration.
A 1 in a particular bit position in the attributes mask indicates to update the configuration with the corresponding bit of the attributes word field.
A 0 in a particular bit position in the attributes mask indicates that the current attribute will be retained.

FIELD	DEFAULT VALUE	FIELD DESCRIPTION
ATTRIBUTES WORD	None	(cont'd) Configuration-Status This field contains the current field value. It is important to realize that the attributes word contains the bit values defining all attributes of the media independent of the attributes mask. This is essential when using removable media on more than one drive/channel type. The bits and their corresponding disk attributes are listed under ATTRIBUTES MASK and described in paragraph 3.5.6.
VERSAdos SECTOR SIZE	None	Configure-Device Configure-Defaults This field defines the number of bytes in a VERSAdos sector. The only value supported is 256. This field is returned by the driver. Configuration-Status This field contains the current field value.
VERSAdos TOTAL SECTORS	None	Configure-Device Configure-Defaults This field is calculated and returned by the driver based on the sectors per track, number of cylinders, and number of surfaces specified for the media. It represents the total number of VERSAdos sectors available on the media.
WRITE TIMEOUT	0	Configure-Device Illegal. An attempt to change this field results in an error; use the Configure-Defaults request instead. Configure-Defaults The number of milliseconds IOS should allow for completion of a Write request before halting I/O. This feature is currently supported by the VM20/VM21 IPC disk controller driver. Configuration-Status This field contains the current field value.

FIELD	DEFAULT VALUE	FIELD DESCRIPTION
READ TIMEOUT	0	Configure-Device Illegal. An attempt to change this field will result in an error; use the Configure-Defaults request instead. Configure-Defaults The number of milliseconds IOS should allow for completion of a Read request before halting I/O. This feature is currently supported by the VM20/VM21 IPC disk controller driver. Configuration-Status This field contains the current field value.
SECTORS/TRACK ON MEDIA	None	Configure-Device Configure-Defaults This field defines the number of physical sectors formatted on a track of media. The number must be accurately given based on the type of media used. This field is used in determining the total number of VERSAdos sectors on the media. Configuration-Status This field contains the current field value.
NUMBER OF HEADS	None	Configure-Device Configure-Defaults This field defines the physical number of heads on the drive and is used to determine the number of recording surfaces for hard disks only. This field is also used in determining the total number of VERSAdos sectors on the media for hard disks only. Configuration-Status This field contains the current field value.

FIELD	DEFAULT VALUE	FIELD DESCRIPTION
NUMBER OF CYLINDERS ON MEDIA	None	Configure-Device Configure-Defaults This field contains the number of cylinders across a single surface of the media. The number must be accurately given based on the type of media. It is also used in determining the total number of VERSAdos sectors on the media. Configuration-Status This field contains the current field value.
INTERLEAVE FACTOR	1	Configure-Device Configure-Defaults This field defines the number of physical sectors between logically staggered sector numbers on a track. An interleave factor greater than one is used when the disk controller cannot transfer contiguous sectors. This field is actively used during the format process of the media. Configuration-Status This field contains the current field value.
SPIRAL OFFSET	None	Configure-Device Configure-Defaults This field defines a vertical stagger of physical sector numbers on a track for a given surface. The spiral offset value is the number of sectors to skip when selecting the next surface in a vertical step. This field is actively used during the format process of the media. Configuration-Status This field contains the current field value.

FIELD	DEFAULT VALUE	FIELD DESCRIPTION
PHYSICAL SECTOR SIZE ON MEDIA	None	Configure-Device Configure-Defaults This field defines the number of bytes in a physical sector on the media. Currently, the only valid values are 128 and 256 bytes per sector corresponding to single data density and double data density, respectively. These values must be defined to be consistent with the data density bit in the attributes word. The driver returns an error if there is a conflict. Configuration-Status This field contains the current field value.
STARTING HEAD NUMBER ON DRIVE	0	Configure-Device Configure-Defaults This field defines the starting head number for the logical unit for dual volume Storage Module Drives (SMD) when supported by the disk controller. Configuration-Status This field contains the current field value.
NUMBER OF CYLINDERS ON DRIVE	None	Configure-Device Configure-Defaults This field defines the maximum number of cylinders on the media that the drive unit can support. For non-removable media, this field is meaningless. Configuration-Status This field contains the current field value.
PRECOMPENSATION CYLINDER NUMBER	None	Configure-Device Configure-Defaults This field defines the cylinder number of the media at which precompensation begins when supported by the disk controller. Unless specifically defined by the drive manufacturer, a value equal to one-half the number of cylinders on the media is satisfactory. Configuration-Status This field contains the current field value.

FIELD	DEFAULT VALUE	FIELD DESCRIPTION
NUMBER OF PHYSICAL SECTORS PER TRACK ON DRIVE	None	Configure-Device Configure-Defaults This field defines the maximum number of physical sectors per track on the media that the drive unit can support. This field is not supported by any of the current disk drivers. Configuration-Status This field contains the current field value.
STEPPING RATE CODE	0	Configure-Device Configure-Defaults This field defines the stepping rate code of the head on the drive unit when supported by the disk controller. The code is translated into a stepping rate in milliseconds by the driver or disk controller for the following disk types:

	Step Rate Code	Winchester Hard Disks	5-1/4 inch Floppy Drive	8-inch Floppy Drive
Default	000	0 msec	12 msec	6 msec
	001	6 msec	6 msec	3 msec
	010	10 msec	12 msec	6 msec
	011	15 msec	20 msec	10 msec
	100	20 msec	30 msec	15 msec

Configuration-Status

This field contains the current field value.

REDUCED WRITE CURRENT CYLINDER NUMBER	None	Configure-Device Configure-Defaults This field defines the cylinder number of the media at which reduced write current begins when supported by the disk controller. Unless specifically defined by the driver manufacturer, a value greater than the number of cylinders should be specified to indicate reduced write current precompensation is not used. Otherwise the precompensation begins at cylinder zero. Configuration-Status This field contains the current field value.

FIELD	DEFAULT VALUE	FIELD DESCRIPTION
ECC DATA BURST LENGTH	None	Configure-Device Configure-Defaults This field defines the number of bits to correct for an ECC error when supported by the disk controller. Configuration-Status This field contains the current field value.

3.5.6 Disk Attributes Word Field

A detailed description of the disk attributes word field follows.

ATTRIBUTE	BIT	DEFAULT VALUE	BIT DESCRIPTION
Data density	0	0	Used to determine the encoding method of a track of data. This is determined by the formatting method. This bit must be set to be consistent with the parameter block sector size field. 0 = FM encoding (single data density, 128 bytes/sector) 1 = MFM encoding (double data density, 256 bytes/sector)
Track density	1	0	Used to define the density across a recording surface. This usually relates to the number of tracks per inch. 0 = Single density 1 = Double density
Side/diskette	2	0	Used to determine if a single- or double-sided diskette is mounted on a drive. 0 = Single sided diskette 1 = Double sided diskette

ATTRIBUTE	BIT	DEFAULT VALUE	BIT DESCRIPTION
Format	3	0	Defines the type of format on the media. The Motorola format is a nonstandard format. 0 = Motorola format 1 = Standard IBM format
Disk type	4	0	0 = Floppy disk 1 = Rigid disk
Drive data Density Capability	5	0	0 = FM (single data density) drive encoding capability 1 = MFM (double data density) drive encoding capability
Drive track Density Capability	6	0	0 = Single density capability 1 = Double density capability
Embedded servo drive seek	7	0	Causes a seek to occur following a head change (SMD drive only). 0 = Drive does not require a seek when a head switch is performed. 1 = Drive requires a seek when a head switch is performed.
Post-read/ pre-write pre-compensation	8	0	Used to define the pre-compensation cylinder number (when supplied) as post-read or pre-write. 0 = Pre-write 1 = Post-read
Floppy disk size	9	0	Defines the size of the media. 0 = 5-1/4 inch floppy disk 1 = 8-inch floppy disk
Alternate Sector Capability	10	0	Used to determine whether the disk driver supports alternate sector handling. 0 = No support 1 = Supports alternate sector handling

3.5.7 Media Attribute Table

Table 3-9 lists the media files (.SI) from User 9998 (null catalog), the media related attributes values, and a cross-reference displaying which media files can be used with the currently supported disk controllers.

TABLE 3-9. Media Attribute Table

ATT_WORD	$0210	$0010	$0010	$0010	$0010	$0010	$0010	$0010	$0010	$0010	$0010	$0204	$0200	$0F4F	$0010	$0090	$0090	$0090	$0090	$0200	$0200
SECT_PER_TRK	32	32	32	32	32	32	32	32	32	32	32	26	26	16	64	64	64	64	64	26	26
NUM_HEADS	4	2	4	6	6	4	4	6	6	1	1	2	2	2	1	5	2	2	2	2	2
CYL_DISK	256	306	306	360	830	153	153	306	830	823	823	77	77	80	823	823	206	206	624	77	77
SPIRAL_OFF	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0
BYTES_PER_SECTOR	256	256	256	256	256	128	128	256	256	256	256	128	128	256	256	256	256	256	256	256	256
START_HEAD_NUM	0	0	0	0	0	0	0	0	0	0	16	0	0	0	0	16	0	0	2	0	0
CYL_DRIVE	256	306	306	306	830	306	360	306	830	823	823	77	77	80	823	823	206	206	624	77	77
PRE_COMP	128	153	153	153	400	153	128	153	400	0	0	38	38	40	0	0	0	0	0	38	38
SECT_DRIVE	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
STEP_RATE	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R_W_PRECOMP	257	307	307	307	831	307	361	307	831	0	0	44	44	0	0	0	0	0	0	44	44
RWIN1	●	●	●	●	●	●		●	●			●	●	●							
SASI5		●	●	●	●	●		●	●					●							
SASI8	●																				
VM20 Floppy IPC												●	●							●	●
VM21 UDC												●	●		●	●	●	●	●	●	●
VM22												●	●		●	●	●	●	●	●	●
VME315	●	●	●			●	●	●				●	●	●	●	●					
VME320	●	●	●		●	●		●	●					●							

3.5.8 Disk Controller Attribute Table

Table 3-10 is a cross-reference displaying the controller related attributes for the currently supported disk controllers.

TABLE 3-10. Disk Controller Attribute Table

DISK CONTROLLER ATTRIBUTE TABLE	RWIN1	SASI5	SASI8	VM20 floppy IPC	VM21 UDC	VM22	VME315	VME320
CHAN_ID	WIND	SAS5	SAS8	CFD1	CUD1	VM22	M315	M320
ATT_MASK hard disk	0410	0015	0210	N/A	03FF	0490	0110	0210
ATT_MASK floppy disk	001F	0015	0215	03FF	03FF	025F	021F	0215
PAR_MASK	1AF3	02F3	02F3	FFF3	FFF3	5FF3	DBF3	53F3
INTERLEAVE hard disk	1	4	4	n/a	0	1	5	11
INTERLEAVE floppy disk	1	8	13	1	1	1	1	1
ECC_LEN	0	0	0	0	0	0	11	0

3.5.9 Magnetic Tape Configuration Parameter Block

The Magnetic Tape Configuration Parameter Block is shown in Figure 3-8.

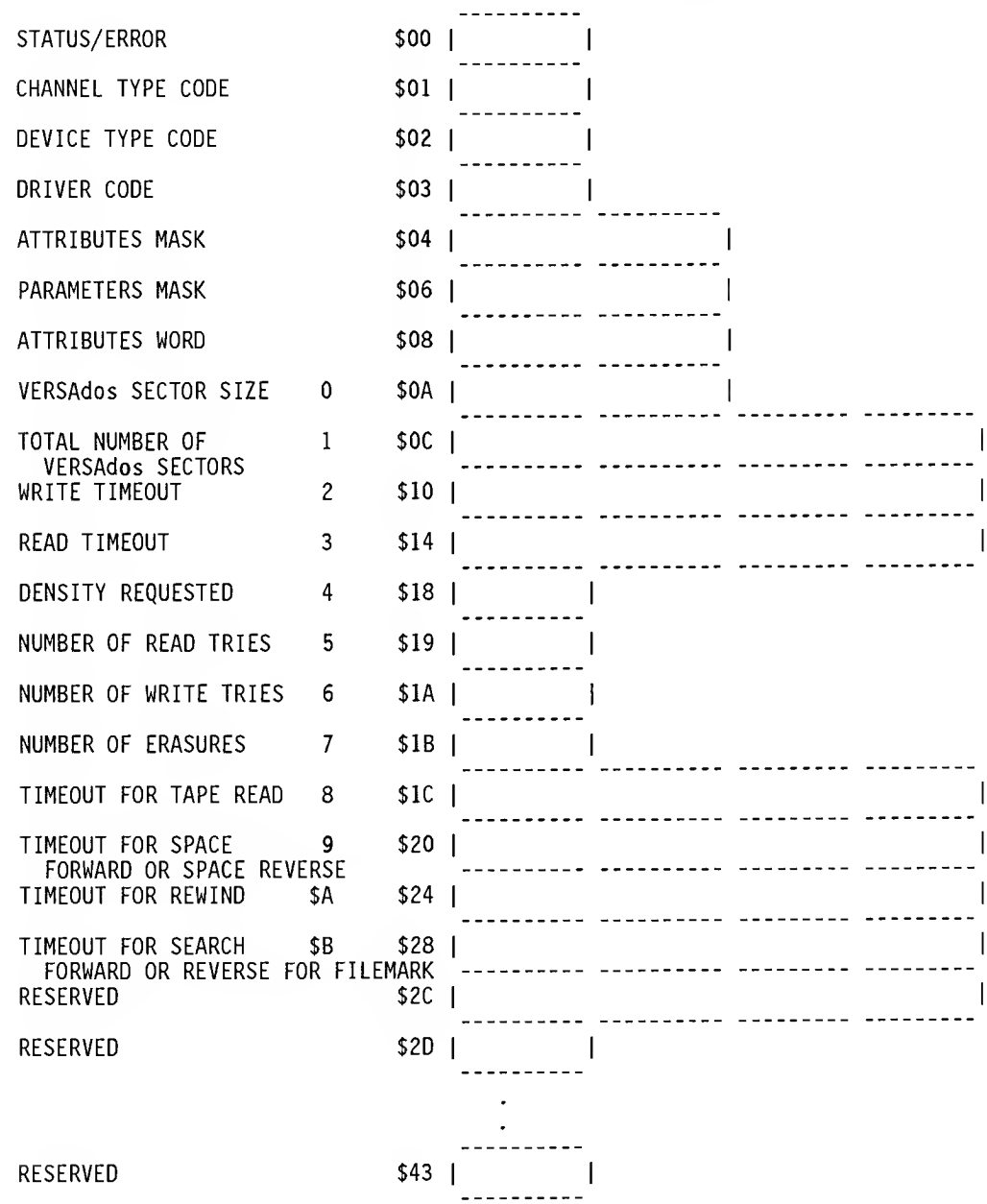


FIGURE 3-8. Configuration IOCB for Magnetic Tape

NOTE: The number in the left number column above is the bit in the parameter-mask field that corresponds to that parameter.

The WRITE TIMEOUT at location $10 is used by the operating system to halt a Write command and by the magnetic tape driver to halt an Erase command.

The READ TIMEOUT at location $14 is used by the operating system to halt any command other than a Write command. To get a more significant (smaller) timeout value for the Read, Space, Rewind, and Search commands for the magnetic tape driver, set the corresponding entries starting with TIMEOUT for TAPE READ to suit the system.

FIELD	DEFAULT VALUE	FIELD DESCRIPTION
STATUS/ERROR		Configuration-Status On return from a request, this field contains the current device status unless the status field in the user's IOCB is nonzero in which case the returned status/error field contains a code identifying the cause of the error.

CONFIGURATION ERRORS

IOSCEC Equate	Code	Device Status
CECCDO	$01	Specified parameter/ attribute invalid for Configure request; use Configure-Defaults request.
CECVAD	$02	Specified parameter/ attribute invalid for driver/channel type.
CECDEN	$41	Unknown density requested.

CHANNEL TYPE CODE — Configuration-Status

On return from a request, this field contains the code for the channel type. If an error is returned in the status/error field this field is reserved for future use.

Code	Type Channel
$24	MVME435 magnetic tape adapter

FIELD	DEFAULT VALUE	FIELD DESCRIPTION
DEVICE TYPE CODE		Configuration-Status If status is returned in the status/error field, this field contains a code identifying the type of device. If an error is returned in the status/error field, this field is reserved for future use. Code — Device Type $05 — Magnetic tape
DRIVER CODE		Configuration-Status On return from a request, this field contains the code for this driver type. If an error is returned in the status/error field, this field is reserved for future use. Code — Driver $07 — Magnetic tape driver on I/O Channel
ATTRIBUTES MASK		Configuration-Status A 1 in a particular bit position indicates that the corresponding bit of the attributes word field should be used to update the configuration. A 0 indicates that the current attribute is retained.
PARAMETERS MASK		Configuration-Status A 1 in a bit position indicates that the driver recognizes the corresponding parameter. A 0 indicates that the parameter is not recognized by the driver.
ATTRIBUTES WORD	0	Configuration-Status This field contains bit values that either determine or reflect the magnetic tape driver's behavior under specific circumstances. Paragraph 3.5.10 describes the values and their correspondence with magnetic tape attributes.
VERSAdos SECTOR SIZE		Not used by magnetic tape.
TOTAL NUMBER OF VERSAdos		Not used by magnetic tape.

FIELD	DEFAULT VALUE	FIELD DESCRIPTION
WRITE TIMEOUT	5000 (5 seconds)	Configuration-Status This field contains the number of milliseconds IOS should allow for Write request completion before halting the I/O. Attempting to change this field using the Configure-Device request results in an error. Use the Configure-Defaults request instead, followed by a Configure request with zeros in the parameters mask.
READ TIMEOUT	360000 (6 minutes)	Configuration-Status This field contains the number of milliseconds IOS should allow for completion of a Read or Output-with-Input request before halting the I/O. An attempt to change this field using the Configure-Device request results in an error. Use Configure-Defaults request instead, followed by a Configure-Device request with zeros in the parameters-mask field.
DENSITY REQUESTED	0	Density for write from loadpoint. 0 = 1600 bpi 1 = 800 bpi
NUMBER OF READ TRIES	3	Number of times the driver tries to read a block before error is sent (0 is equivalent to 1).
NUMBER OF WRITE TRIES	3	Number of times the driver tries to write a block before trying to erase "4" of tape (0 is equivalent to 1).
NUMBER OF ERASURES	1	Number of times the driver erases the tape before reporting an error.
TIMEOUT FOR TAPE READ	5000 (5 seconds)	This field contains the number of milliseconds the driver allows for completion of a tape read. This field can be changed with a Configure-Device request.
TIMEOUT FOR SPACE FORWARD OR SPACE REVERSE	300000 (5 minutes)	This field contains the number of milliseconds the driver allows for completion of a Space-Forward or Space-Reverse command. A Configure-Device request can change this field.

FIELD	DEFAULT VALUE	FIELD DESCRIPTION
TIMEOUT FOR REWIND	300000 (5 minutes)	This field contains the number of milliseconds the driver allows for completion of a Space-Forward or Space-Reverse command. Change field with Configure-Device request.
TIMEOUT FOR SEARCH FORWARD OR REVERSE FOR FILEMARK	300000 (5 minutes)	This field contains the number of milliseconds the driver allows for completion of a Search-Forward-For-Filemark or Search-Reverse-For-Filemark. A Configure-Device request can change this field.

3.5.10 Magnetic Tape Attributes Word Field

ATTRIBUTE	BIT	DEFAULT VALUE	BIT DESCRIPTION
	15-2		Reserved
Density request	1	0	1 = User requests a density for write from loadpoint.
			0 = User does not request a density for write from loadpoint.
	0		Reserved

3.6 CONFIGURATION PARAMETER BLOCK EQUATES

The following defines the symbols for the system equates (constants) that contain the offsets for the fields within the configuration parameter block.

3

FIELD	EQUATE			
	TERMINAL	DISK	PRINTER	MAGNETIC TAPE
ATTRIBUTES MASK	IOSATM	IOSATM	IOSCTP	IOSCTP
CONFIGURATION ERROR CODE	IOSCEC	IOSCEC	IOSCEC	IOSCEC
DATA BLOCK LENGTH	----	----	IOSPLN	IOSMLN
DEVICE TYPE CODE	IOSDTP	IOSDTP	IOSDTP	IOSDTP
DISCARD OUTPUT	IOSDOP	----	----	----
DENSITY REQUEST	----	----	----	IOSDEN
DRIVER CODE	IOSDRC	IOSDRC	IOSDRC	----
ECC DATA BURST LENGTH	----	IOSECC	----	----
END-OF-LINE STRING/CHARACTER	IOSEOL	----	IOSELC	----
ERASURES	----	----	----	IOSERT
INTERLEAVE FACTOR	----	IOSILV	----	----
LINES PER PAGE	IOSRSZ	----	IOSRSZ	----
LOGICAL LINE WIDTH	----	----	IOSLRL	----
NULL PADDING	IOSNLS	----	----	----
NUMBER HEADS	----	IOSHDS	----	----
NUMBER OF SECTORS TRACK-DRIVE	----	IOSPTD	----	----
NUMBER TRACKS	----	IOSTRK	----	----
MEDIA NUMBER TRACKS-DRIVE	----	IOSTRKB	----	----
MEDIA SECTOR SIZE	----	IOSPSM	----	----
PARAMETERS MASK	IOSPRM	IOSPRM	IOSPRM	IOSPRM
PHYSICAL LINE WIDTH	IOSREC	----	IOSREC	----
PRECOMPENSATION CYLINDER #	----	IOSPCOM	----	----
READ TERMINATORS	IOSRTV	----	----	----
REDUCED WRITE CURRENT CYLINDER #	----	IOSEWCC	----	----
READ TIMEOUT	IOSRTO	IOSRTO	----	IOSRTO
READ TRIES	----	----	----	IOSRDT
REPRINT LINE	IOSRLN	----	----	----
SECTOR SIZE	----	IOSREC	----	----
SECTORS/TRACK	----	IOSSPT	----	----
SPIRAL OFFSET	----	IOSSOF	----	----
STARTING HEAD NUMBER	----	IOSSHD	----	----
STATUS/ERROR	IOSDST	IOSDST	IOSDST	IOSDST
STEPPING RATE	----	IOSSRTD	----	----
TERMINATOR-CLASS	IOSTRC	----	----	----
TOTAL SECTORS	----	IOSRSZ	----	----
WRITE TIMEOUT	IOSWTO	IOSWTO	IOSWTO	IOSWTO
WRITE TRIES	----	----	----	IOSWRT
XOFF CHARACTER	IOSXOF	----	----	----
XON CHARACTER	IOSXON	----	----	----
TIMEOUT FOR TAPE READ	----	----	----	IOSRDTO
TIMEOUT FOR SPACE FORWARD OR REVERSE	----	----	----	IOSSPTO
TIMEOUT FOR REWIND	----	----	----	IOSRWTO
TIMEOUT FOR SEARCH FORWARD	----	----	----	IOSSRTO

CHAPTER 4

SUPPORTED DEVICES

4.1 INTRODUCTION

All input/output requests are made via directive Input/Output Services (IOS) calls. This chapter discusses the functional aspects of the devices and direct-access files supported by VERSAdos. Specific device-dependent information includes functions supported, status returned, and formatting performed.

All devices and files support ASCII formatting, wait, I/O, and sequential access, unless otherwise noted in the individual driver or file handler description. The supported attributes listed with each description are in addition to those previously listed.

A device code, a number between 20 and 254, defines all VERSAdos-supported devices. Additionally, device codes numbered between 0 and 19 differentiate file types.

4.2 CONTIGUOUS FILES

The following paragraphs describe contiguous files.

4.2.1 Devices Supporting Contiguous Files

Contiguous files are supported on all devices supported by hard disk drivers and/or floppy disk drivers. The device code is 0.

4.2.2 Supported Attributes

Attributes of contiguous files are:

- Read
- Write
- Binary
- Wait
- Random
- Connection wait
- Image
- Block access
- Rewind

4.2.3 Functional Description

Contiguous files may be randomly or sequentially accessed. These two access methods may be mixed without closing and reassigning the file. Reference to the current record pointer will access sectors. For sequential access, current record pointer value, as left by the previous operation, accesses the file. For random access, the user specifies the current record pointer to be used. When a contiguous file is assigned, the current record pointer is set to the beginning of the file unless write access with position at end is requested. Here, the current record pointer is positioned to the logical End-Of-File (EOF).

Read: The formatted/image and record/block access options are ignored. All reads are image with block access. The requested data is directly read from the disk into the user-specified buffer. Reading starts at the specified RRN in the IOS call (random access) or can be determined by the current record pointer. For a Read-Current request, the current record pointer value is used. For a Read-PROPR request, the current record pointer minus one is used. For a Read-Next request, the sector to read is determined by current record pointer plus the number of sectors transferred on the previous request. On completion of the request, current record pointer is set to the first sector read. All reads begin on a sector boundary and end when the specified number of bytes have been transferred or the logical EOF is encountered. The buffer size specified must be a multiple of 256.

Write: Same as Read except the data is transferred from the user specified buffer to the disk. The number of bytes transferred must be an integral multiple of 256.

Rewind: The current record pointer is set to the beginning of the file so that the Read-Next request retrieves the first sector in the file.

Status Definition:

STATUS	MEANING
$00	Normal completion.
$82	Invalid function. Sequential request for CURRENT on first request or request PRIOR when current record pointer is at beginning of file. Write request for file assigned for read access only, or Read request for file assigned for write access only.
$84	Invalid data buffer address. Data buffer not multiple of 256 bytes.
$87	Write-protected file.
$C2	End-of-file. Attempted to read starting beyond the EOF or attempted write beyond the EOF.
$E1-$ED	Device-dependent error.

4.3 SEQUENTIAL FILES

The following paragraphs describe sequential files.

4.3.1 Devices Supporting Sequential Files

Sequential files are supported on all devices supported by hard disk drivers and/or floppy disk drivers. The device code is 1.

4.3.2 Supported Attributes

Attributes of sequential files are:

- Read
- Write
- Update record
- Block and logical record access
- Binary
- Wait
- Random
- Connection wait
- Image
- Rewind
- Position

4.3.3 Functional Description

Sequential files may be randomly or sequentially accessed. In addition, block and record access is allowed and may be mixed without closing and reassigning the file. Referring to the current record pointer accesses records if doing record I/O, sectors if doing block I/O. For sequential access, current record pointer value as left by the previous operation accesses the file. For random access, the user specifies the current record pointer to be used.

When a sequential file is first allocated, no data space exists. Therefore, it is necessary to write blocks or records sequentially before reading or writing randomly to the file.

When a sequential file is assigned, the current record pointer is set to the beginning of the file unless position at end is requested. Here, the current record pointer is positioned to the EOF.

The shared segment option is requested at assignment and must be selected if performing block writes or if block and record accesses are intermixed. This option causes FHS to allocate a memory segment whose size is determined by the file's data block size and grants shared access of this segment to the user. If the user selects this option, all block transfers must use this shared segment.

If the shared segment option is not selected and block reads are requested, the user must provide a data buffer large enough to accommodate the data block size. If only record I/O is requested, and no shared segment was allocated, a data block segment is allocated by FMS on the first I/O request.

Read block: The requested data is directly read from disk into the user-specified buffer (must be the shared segment if one exists). One data block is transferred for each read request. Reading starts at the RRN (sector number) specified in the IOS call (random access). An "invalid random record number" error occurs if the RRN does not correspond to the first sector of a data block. Here, the data block is still transferred and the caller's IOS block is modified so that the logical sector number corresponding to the first sector in the data block is returned in the random record number field. Read-Next, Prior, and Current requests access the next, prior and current data block, respectively, based on the value of the current record pointer.

Write block: The data buffer must be the shared segment allocated at assignment. The data is transferred directly from the shared segment to disk. Random and sequential features are identical to read block. An error occurs if a block Write follows record Write and does not reference the same data block that the record resided in. If a random Write is requested and the RRN specified does not correspond to the first sector of a data block, an error occurs and the transfer is not done.

Read logical record: The data block containing the logical record specified by the RRN passed (random access) or determined by the current record pointer (sequential access) is read into a data block segment (unless it is already in memory). The requested number of bytes is moved from the data block to the user-specified buffer. The transfer ends when the entire record is transferred or the user's buffer is full. If the record is fixed length, the data is transferred as is to the user buffer. If the record is variable length and the formatted ASCII options are set on the I/O request and the file is not a spooler file, spaces are expanded as the record is transferred to the user buffer.

The current record pointer is set to the record transferred. Read-Next, Prior, and Current record requests are supported.

Write logical record: Write record requests are only valid when being appended to the end of file; otherwise use Update. The data block containing the last record in the file is transferred into memory. After the last record, the record passed in the user buffer is transferred into the data block. For fixed length records, the record length determined by the user buffer start and end addresses must equal the record length. Fixed length records are moved as is to the data buffer. If the record is variable length, and the formatted ASCII options are set on the I/O request and the file is not a spooler file, spaces are compressed as the record is transferred to the data buffer. For spooler files, the transfer is always image and the options word for the IOS parameter block is inserted at the beginning of the record. The current record pointer is set to the record transferred. The only sequential Write request that could follow would be Next. For multiple write assignments, a Write-Next request always appends a record to the end of the file, regardless of where the last request left the current record pointer.

Update record: The data block containing the logical record specified by the RRN passed (random access) or determined by the current record pointer (sequential access), is read into data block segment (unless it is already in memory). The length of the user record must be the same as the existing record. The record is transferred to the data block. Set the current record pointer to the record updated. To update a record, the file must be assigned EREW.

Rewind: Set the current record pointer to the beginning of the file.

Position: Set the current record pointer in the same way as an I/O request updating the current record pointer. Position to end of file is requested by doing a random logical record position with the RRN equal to -1. The current record pointer value is returned in the user's RRN field. Positioning to EOF returns the number of records in the file minus one. The requested record offset in the data block is returned in the user's length field; valid only if record I/O requested.

Status Definition:

STATUS	MEANING
$00	Normal completion.
$0C	Insufficient system space. No room to allocate data buffer for record I/O.

$82 Invalid function. Sequential request for Current on first I/O request or request Prior when current record pointer at beginning of file. Block Write for different data block than that accessed for record Write that preceded call. Intermixing block and record access without shared segment. Block Write was requested and shared segment does not exist. Delete-Record request. Random key request. Position request with block I/O. Update-Record request and file not assigned EREW access permission.

$84 Invalid data buffer address. Data buffer length for fixed length records not equal to record length. Data buffer length for variable length record write larger than data block size. Data buffer addresses for record I/O conflict with address of shared segment (cannot use shared segment for record data). For Update-Record request, data buffer length not same size as existing record. If shared segment exists and block I/O requested, data buffer starting address not same as that specified at assignment, data buffer length not equal to data block size for block Read, or data buffer length larger than data block size for block Write. For block Write request, data buffer length not multiple of record length for fixed length record files or too short to accommodate all variable length records. For a Write/Update-Record request with space compression, user's record length greater than 256 bytes.

$85 Invalid random record. On block I/O, RRN specified does not correspond to first sector in data block.

$87 Write-protected file. Write or Update-Record selected for write-protected file.

$C1 Buffer overflow. User data buffer for record Read request not large enough to accommodate entire record. For a Read request with space expansion, expanded record length greater than 256 bytes.

$C2 End-of-File. Attempted to do Read or Update-Record starting beyond the end of the file or attempted Write starting beyond EOF plus one.

$C8 FAB/data block conflict. With a shared segment, user destroyed some data in segment.

$C9 Record does not exist. Update-Record request specified record that did not exist.

$CA Record already exists. Write record request specified record that already existed.

$CB Record overflow. With a shared segment, user destroyed some data in segment, causing a variable length of record to go beyond the end of the data block.

$CD Insufficient disk space. No space available on volume for new FAB or data block.

4.4 INDEXED SEQUENTIAL FILES (NO DUPLICATE KEYS)

The following paragraphs describe indexed sequential files (no duplicate keys).

4.4.1 Devices Supporting Indexed Sequential Files Without Duplicate Keys

Indexed sequential files are supported by hard disk drivers and/or floppy disk drivers. The device code is 2.

4.4.2 Supported Attributes

Indexed sequential files attributes are:

Read
Write
Update record
Delete record
Block access
Logical record access
Random record access
Read record with or without key
Binary
Wait
Random
Connection wait
Image
Rewind
Position

4.4.3 Functional Description

Indexed sequential files support all the functions described for sequential files (refer to paragraph 4.3). The only difference is that the Update-Record function allows the new record to be a different length than the old record, as long as it is large enough to accommodate the key field. The key in the new record must be the same as the key in the old record.

On a record Read request (either logical record or random key), the "return key with record" option is examined. If the option is not selected, the data returned to the user does not include the key. This makes it possible to have a zero data transfer length if the record contained only a key and no other data. If this option is selected, the entire record with key is returned to the user.

For logical record writes, records must be appended to the end of the file. In addition, the key of the new record must be greater than the key of the current last record.

Additional functions supported:

Read random key: The key of the record to be read is passed in the user's data buffer. If a record with the specified key exists, it is transferred to the user's buffer in the same way as logical record reads. If no record with the requested key exists, a "record does not exist" error is returned.

Write random key: The record key to be written is passed in the user's data buffer, along with the rest of the record. If a record already exists with the key, a "record already exists" error is returned. Otherwise, the record is inserted in the file in ascending sequence by key, with spaces being compressed if the ASCII formatted options were selected for a variable length record.

Delete record: The record key to be deleted is passed in the user's data buffer. If a record with the specified key exists, it is deleted from the file.

Status Definition: Certain device-dependent errors return two sector numbers.

STATUS	MEANING
$00	Normal completion.
$0C	Insufficient system space. No room to allocate data buffer for record I/O.
$82	Invalid function. Sequential request for Current on first I/O request or after record Delete or request Prior when current record pointer at beginning of file. Block Write for different data block than that accessed for record Write that preceded call. Intermixing block and record access without shared segment. Block Write requested and shared segment does not exist. Position request for block I/O.
$84	Invalid data buffer address. Data buffer length for fixed length records not equal to record length. Data buffer length for variable length record Write larger than data block size or smaller than key size (all requests). Data buffer addresses for record I/O conflict with address of shared segment. If shared segment exists and block I/O requested, data buffer starting address not same as that specified at assignment, data buffer length not equal to data block size for block Read, or data buffer length larger than data block Write. For block Write request, data buffer length not multiple of record length for fixed length record files or too short to accommodate all variable length records. For a Write/Update-Record request with space compression, user's record length greater than 256 bytes.
$87	Write-protected file. Write, Update-Record or Delete-Record selected for write-protected file.

$C1 Buffer overflow. User data buffer for record Read request not large enough to accommodate entire record. For a Read request with space expansion, expanded record length greater than 256 bytes.

$C2 End-of-File. Attempted to do Read or Update-Record starting beyond the end-of-file or attempted logical record Write or block Write starting beyond EOF plus one.

$C8 FAB/data block conflict. With a shared segment, user destroyed some data in segment.

$C9 Record does not exist. Update-Record or Delete-Record request or random key Read request specified record that did not exist.

$CA Record already exists. Write record request specified record that already existed.

$CB Record overflow. With a shared segment, user destroyed some data in segment, causing a variable length record to go beyond the end of the data block.

$CC Key error, FAB/key conflict. With a shared segment, user destroyed some data in segment, causing keys to be out of sequence.

$CD Insufficient disk space. No space available on volume for new FAB or data block.

4.5 INDEXED SEQUENTIAL FILES (DUPLICATE KEYS ALLOWED)

The following paragraphs describe indexed sequential files (duplicate keys allowed).

4.5.1 Devices Supporting Indexed Sequential Files With Duplicate Keys

Indexed sequential files (duplicate keys allowed) are supported by hard disk drivers and/or floppy disk drivers. The device code is 1 for the hard disk driver and 2 for the floppy disk driver.

4.5.2 Supported Attributes

Same as for indexed sequential files (no duplicates). Refer to paragraph 1.3.2.

4.5.3 Functional Description

Similar to description in paragraph 4.4.3. The main difference is that no error is given if it is a Write request and the key of the new record is the same as the key in an existing record. For random key access, if more than

one record exists with the same key, the first record is always accessed. The remaining records with the same key can be accessed by logical record access of Retrieve-Next.

Logical record writes need not always append to the end of file. The exception is a request to Write-Prior or Write-Next such that the key of the new record is the same as the key of the current record.

Status Definition: Refer to paragraph 4.4.3.

4.6 INTERACTIVE TERMINAL

The following paragraphs describe the attributes supported for an interactive terminal, and provide a function description.

4.6.1 Supported Attributes

Supported attributes for terminal devices are:

Read
Write
Output with input
Wait
Proceed
Connection wait
Image
Interactive
Halt I/O
Variable length records

4.6.2 Functional Description (Teletype Configuration)

Read image: Incoming characters are masked to 7 bits if 7 bits per character transmission was selected in the configuration.

Data is read into the buffer until termination occurs. Normal termination can occur in one of three ways:

a. The received character matches a READ TERMINATOR VALUE.

b. The received character falls into the TERMINATOR CLASS.

c. The buffer is full and the TERMINATE ON BUFFER FULL attribute is set.

If the first or second method caused termination, the terminator character is placed in the buffer and is included in the LENGTH OF DATA TRANSFER.

The following characters have special interpretations when received:

XOFF	)
XON	) Refer to descriptions of these parameters in
BREAK EQUIVALENT	) Chapter 3.
DISCARD OUTPUT	)
$00	NUL (discarded unless PASS NULS attribute is set)

Read formatted: Similar to read image with the following exceptions:

The terminator character (if there is one) is not included in the LENGTH OF DATA TRANSFER byte count, although it is still placed in the buffer.

The special characters are:

XOFF	)
XON	)
BREAK EQUIVALENT	) Refer to descriptions of these parameters in
DISCARD OUTPUT	) Chapter 3.
REPRINT LINE	)
CANCEL LINE	)
$00	NUL (always discarded)
$08	BS (deletes last character entered)
$01-$07, $09-$1F	Other control characters ($01 echoes as A, etc.)
$7F	DEL (like backspace)
$80-$FF	Invalid (character is discarded; bell is sounded)

The terminator character is not echoed. Whenever normal termination occurs, the EOL string is echoed to the terminal.

Write image: All characters in the buffer are sent to the terminal.

Write formatted: All characters in the buffer up to but not including the first $0D (CR) are sent to the terminal. If the number of characters written is not an exact multiple of the configured line length, the EOL string is sent.

Output with input: There are four varieties of output with input, resulting from the choice of formatted versus image modes for the output and for the input. A Write operation followed by a Read operation is performed. The LENGTH OF DATA TRANSFER byte count reflects the input, rather than the output.

Halt I/O: Abnormally terminates any of the above requests outstanding.

NOTE

A Halt-I/O command, a break received during I/O, a parity error, a framing error, or an overrun condition will abnormally terminate any I/O and cause a partial reset of the port.

Configure-Device)
Configure-Defaults) These commands are described in detail in Chapter 3.
Configuration-Status)

Status Definition:

STATUS	MEANING
$00	Normal completion
$82	Invalid function
$84	Invalid data buffer address
$C1	Buffer overflow (read only)
$C6	Break condition
$D1-$EF	Device-dependent error

4.7 LINE PRINTER

The following paragraphs define the supported line printers and describe the supported line printer attributes.

4.7.1 Supported Devices

The device codes are returned in the high-order byte of the FHS parameter block options field for a Retrieve-Attributes call. Refer to paragraph 2.3.8.

DEVICE	DEVICE CODE
Low speed (60-200 lpm) line printer	90
High speed (600 lpm) line printer	91
Low speed (60-200 lpm) line printer	95

NOTE

These codes are returned on an FHS Retrieve-Attributes call and are shown only for backward compatibility. The Configuration-Status request should be used to retrieve a device code from the current configuration values.

4.7.2 Supported Options

Wait
Proceed
Connection wait
Formatted
Image

Variable length records are supported.

4.7.3 Functional Description

The Halt-I/O command function and the Write, Configure-Device, Configure-Defaults, and Configuration-Status data transfer requests are supported.

Write formatted: The user buffer is output until a CR is found, the buffer is empty, or the logical record width is reached with the Wraparound/Truncate attribute set. Driver operation is affected by various configuration parameters and attributes. Refer to Chapter 3.

Write image: Data is transmitted exactly as found in the user's buffer. The system does not ensure that the data is printed later or that the paper is properly moved. The user should be familiar with the characteristics of the device being used. Except for the WRITE TIMEOUT and FORMFEED AFTER ASSIGN attributes, all configuration parameters and attributes are ignored in this mode.

Halt-I/O: Any outstanding Write requests are terminated.

Configure-Device
Configure-Defaults
Configuration-Status: These requests are described in Chapter 3.

Status Definition:

STATUS	MEANING
$00	Normal completion
$82	Invalid function
$84	Invalid data buffer address
$D1-$EF	Device-dependent error

4.8 HARD DISK

The supported hard disk devices, their supported attributes, and a functional description are provided in the following paragraphs.

4.8.1 Supported Devices

Devices configured on:

Universal Disk Controller (UDC)

Winchester controller board (RWIN1)

MVME420 interface board
using an SA1403 or DTC520A controller boards

MVME315 disk controller board

MVME320 disk controller board

VM22 disk controller

4.8.2 Supported Attributes

Supported for hard disks are:

Read
Write
Block access
Binary
Wait
Proceed
Random
Connection wait
Retry
Alternate track
Image

Image attributes are supported; sequential access and formatted are not supported.

4.8.3 Functional Description

Read: The data is read from the disk starting at the sector specified by the RRN passed until the buffer is full or until the end of the disk is reached. If the RRN specifies a sector beyond the end of the disk, an error is returned.

Write: Data is written from the buffer to the disk, starting at the sector specified in the RRN field until the buffer is empty. If an attempt is made to write beyond the end of disk, an error is returned. Here, no data is transferred.

Format: Formats the disk or track containing the PSN. If the format alternate track option is set, the PSN is mapped to an alternate track.

Errors on data transfers cause the operation to be retried before returning an error status unless directed not to retry.

All data transfers start and end on a sector boundary. Each sector is 256 bytes.

Status Definition: Certain device-dependent errors return a sector number.

STATUS	MEANING
$00	Normal completion.
$82	Invalid function. Sequential access requested. Format of file requested or volume/device not assigned EREW.
$84	Invalid data buffer address. Data buffer not multiple of 256 bytes.
$85	Invalid random record. Request starts beyond end of disk.
$88	Invalid disk format.
$C3	End of volume. Write request goes beyond end of disk. No data is transferred. Read request starts before end of disk but request is to transfer data beyond the end of disk. The data length returned indicates the actual number of bytes transferred.
$D1-$EF	Device-dependent error.
$F1-$F9	Channel-dependent error.

4.9 FLOPPY DISK

The supported floppy disk devices, their supported attributes, and a functional description are provided in the following paragraphs.

4.9.1 Supported Devices

Devices configured on:

Floppy Disk Controller (FDC)

Universal Disk Controller (UDC)

Winchester controller board (RWIN1)

MVME420 interface board
using an SA1403 or DTC520A controller boards

MVME315 controller board

MVME320 controller board

VM22 disk controller

4.9.2 Supported Attributes

The floppy disk supports the same attributes as the hard disk. Refer to paragraph 4.8.2 for a complete description.

4.9.3 Functional Description

The floppy disk functions are similar to those of the hard disk. Reference paragraph 4.8.3 for a description of the floppy disk Read, Write, and Format functions.

4.10 MAGNETIC TAPE

The supported magnetic tape attributes and a functional description are provided in the following paragraphs.

4.10.1 Supported Attributes

Read
Write
Binary
Image
Halt I/O
Filemark

4.10.2 Functional Description

Read: A block of data (no limit on length) is read from the tape to a memory buffer designated by the user. If the block contains more bytes than can fit in the user buffer, the buffer overflow message is returned.

Write: A block of data of any length is written to the tape.

Configure-and-Mount Command

Parameter block (IOCB):

```
IOSBLK:  DC.B    $01      Request code
         DC.B    $80      Function specification
         DC.W    $0000    Options field
         DC.B    $00      Status
         DC.B    LUN      Logical unit
         DC.W    0        Reserved
         DC.L    CSB      CSB address
         DC.L    0        Reserved
         DC.L    0        Reserved
         DC.L    0        Reserved
         DC.L    0        Reserved
```

The Configure-and-Mount command records the configuration requested by the user in the CSB. If the requested configuration is legal, the tape is rewound, if necessary, to bring the tape to loadpoint, and the first block on the tape is read in an attempt to determine:

a. Tape density

b. Whether the transport density can be controlled by the software

If the transport density is not software selectable, but must be selected by pushing a button on the front panel of the transport, and if the user has selected the transport density 800 bpi, then tape density as recorded in the status word may not be accurate.

If a previous Configure-and-Mount command has succeeded (which can happen if there is a tape on the transport when the system is booted), then succeeding Configure-and-Mount commands are rejected until a Configure-and-Dismount command has been received.

Configure-Only Command

Parameter block (IOCB):

```
IOSBLK:  DC.B     $01       Request code
         DC.B     $80       Function specification
         DC.W     $0001     Options field
         DC.B     $00       Status
         DC.B     LUN       Logical unit
         DC.W     0         Reserved
         DC.L     CSB       CSB address
         DC.L     0         Reserved
         DC.L     0         Reserved
         DC.L     0         Reserved
         DC.L     0         Reserved
```


The Configure-Only command records the configuration requested by the user in the CSB. The tape does not move. If the requested configuration is legal, it takes effect when the next command is sent to the tape drive. (Figure 3-7 describes the CSB block.)

After a tape has received a Configure-and-Mount command, any number of Configure-Only commands are accepted. The Configure-Only command is used to change any of the following attributes or parameters. The Configure-and-Mount command may also be used to change these attributes and parameters.

a. Requesting a certain density for write from loadpoint.

   To request 1600 bpi density, for example, set bit 1 of the attributes mask and the attributes word, set bit 4 of the parameter mask, and put 0 into the DENSITY REQUESTED field of the CSB (offset $18).

```
ATTRIBUTES MASK       $02
PARAMETERS MASK       $10
ATTRIBUTES WORD       $02
DENSITY REQUESTED       0
```

   To request 800 bpi density, set bit 1 of the attributes mask and the attributes word, set bit 4 of the parameter mask, and put 1 into the DENSITY REQUESTED field of the CSB (offset $18).

```
ATTRIBUTES MASK       $02
PARAMETERS MASK       $10
ATTRIBUTES WORD       $02
DENSITY REQUESTED       1
```

b. Changing the number of read tries before an error message is sent back.

To change the number of read tries to 6, for example, set bit 5 of the parameters mask and put 6 into the NUMBER OF READ TRIES field of the CSB (offset $19).

```
ATTRIBUTES MASK          $00
PARAMETERS MASK          $20
NUMBER OF READ TRIES       6
```

Putting a 0 into the number of read tries defaults to 1 read try.

c. Changing the number of write tries before an erasure of the tape is done.

To change the number of write tries to 2, for example, set bit 6 of the parameters mask and put 2 into the NUMBER OF WRITE TRIES field of the CSB (offset $1A).

```
ATTRIBUTES MASK          $00
PARAMETERS MASK          $40
NUMBER OF WRITE TRIES      2
```

Putting a 0 into the number of write tries defaults to 1 write try.

d. Changing the number of erasures after write tries before an error message is returned.

To change the number of erasures to 4, for example, set bit 7 of the parameters mask and put 4 into the NUMBER OF ERASURES field of the CSB (offset $1B).

```
ATTRIBUTES MASK        $00
PARAMETERS MASK        $80
NUMBER OF ERASURES       4
```

Configure-and-Dismount Command

Parameter block (IOCB):

```
IOSBLK:  DC.B     $01       Request code
         DC.B     $80       Function specification
         DC.W     $1000     Options field
         DC.B     $00       Status
         DC.B     LUN       Logical unit
         DC.W     0         Reserved
         DC.L     CSB       CSB address
         DC.L     0         Reserved
         DC.L     0         Reserved
         DC.L     0         Reserved
         DC.L     0         Reserved
```

The Configure-and-Dismount command dismounts the tape:

a. No Read, Write, or other command will be accepted for the tape drive, except Configure and Status commands.

b. When the current assignment to the tape drive is dissolved, the configuration reverts to default configuration.

It is recommended to issue a Dismount command when a tape is removed from the transport, followed by a Configure-and-Mount command when a new tape is placed on the transport, to determine the density of the new tape.

Configure-Defaults Command

Parameter block (IOCB):

```
IOSBLK:  DC.B     $80       Request code
         DC.B     $02       Function specification
         DC.W     $0000     Options field
         DC.B     $00       Status
         DC.B     LUN       Logical unit
         DC.W     0         Reserved
         DC.L     CSB       CSB address
         DC.L     0         Reserved
         DC.L     0         Reserved
         DC.L     0         Reserved
         DC.L     0         Reserved
```

The Configure-Defaults command changes the default values for any of the configuration attributes or parameters. The only way to change WRITE TIMEOUT or READ TIMEOUT is to use the Configure-Defaults command.

To change the WRITE TIMEOUT to 1000 milliseconds (one second), for example, set bit 2 of the parameters mask and put 1000 in the WRITE TIMEOUT field of the CSB (offset $10).

```
ATTRIBUTES MASK     $00
PARAMETERS MASK     $04
WRITE TIMEOUT       1000
```

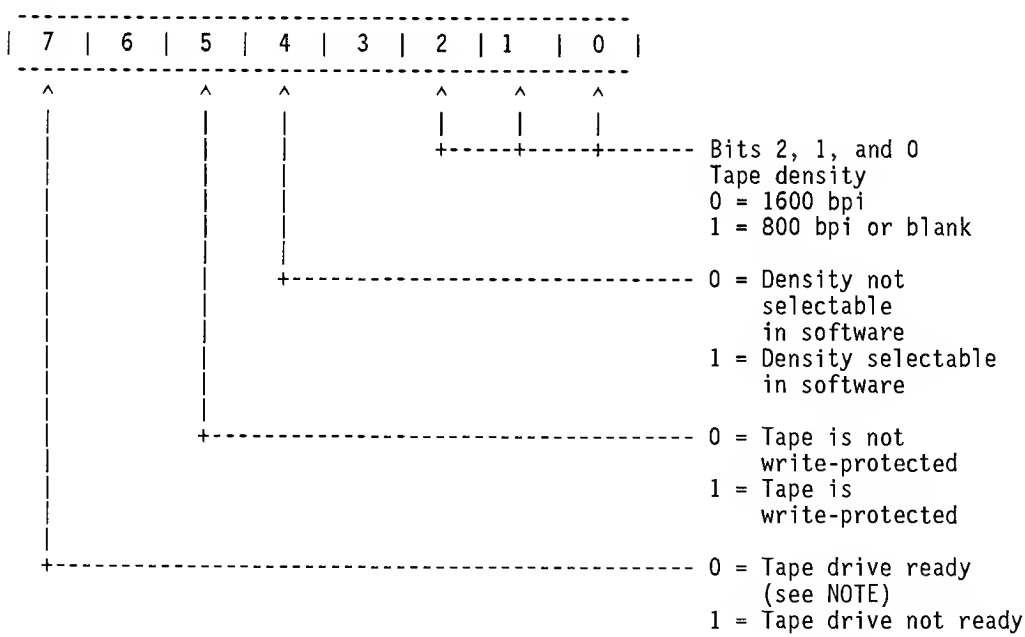
Configuration-Status Command

Parameter block (IOCB)

```
IOSBLK:  DC.B     $01       Request code
         DC.B     $40       Function specification
         DC.W     $0000     Options field
         DC.B     $00       Status
         DC.B     LUN       Logical unit
         DC.W     0         Reserved
         DC.L     CSB       CSB address
         DC.L     0         Reserved
         DC.L     0         Reserved
         DC.L     0         Reserved
         DC.L     0         Reserved
```

The Configuration-Status command returns the status and current configuration of the tape drive.

STATUS byte:



NOTE: This bit is set if the initial Configure-and-Mount command from IOS fails because of a bad parameter or attribute. It is also set when a Configure-and-Dismount command with all valid parameters and attributes is received.

Write-Filemark: A filemark is written to the tape.

Erase: Four inches of tape is erased.

Space-Forward: A block of data is spaced over on the tape. If the block was a filemark, the end-of-file message is returned.

Space-Reverse: Similar to Space-Forward, but in the reverse direction.

Search-Filemark Forward: The tape is advanced until a filemark is detected, the end of tape is reached, or the transport has found a length of blank tape and has timed out. A normal completion message is returned if a filemark is detected.

Search-Filemark Reverse: Similar to Search-Filemark-Forward, but in reverse direction.

Rewind: The tape is rewound to loadpoint. If successful, a normal completion message (00) is returned.

Halt-I/O: Terminates any of the above requests.

SUMMARY OF COMMANDS FOR MAGNETIC TAPE

	CODE 0 COMMANDS		CODE 1 COMMANDS			
	READ	WRITE	REWIND	CONFIGURE	STATUS	HALT I/O
Code	$00	$00	$01	$01	$01	$01
Function	$01	$02	$02	$80	$40	$10
Options	for Configure command			$0000 (mount) $0001 (configure only) $1000 (dismount)		

	CODE 40 COMMANDS					
	ERASE	WRITE FILEMARK	SEARCH FORWARD FOR FILEMARK	SEARCH REVERSE FOR FILEMARK	SPACE FORWARD	SPACE REVERSE
Code	$40	$40	$40	$40	$40	$40
Function	$02	$04	$06	$07	$08	$09

4.10.3 Error Messages

$00	Normal completion, no errors.	
82	Invalid function.	
88	Configuration error.	
C1	Buffer overflow.	
	Read:	User buffer too small.
C2	End-of-file.	Filemark detected on Read or Space-Forward or Space-Reverse.
C3	End of volume.	End of tape detected.
	Read:	Data past end of tape is inaccessible.
	Write:	Some data may be written. It is best to Space-Reverse and Write the block to another tape. Check the length field in parameter block to see how many bytes were actually transferred.
D4	Device not mounted	No Configure-and-Mount has been successful since last Dismount command.
D5	Beginning of volume	Beginning of tape (loadpoint) detected. Tape was at loadpoint when a reverse type command was received (except Rewind), or loadpoint was detected at the completion of the command.
D7	Tape already mounted.	To change configuration on tape that is already mounted, do a Configure-Only command.
E1	Device not ready.	
E3	Error on Read or Write.	
	READ:	It is best to Space-Reverse.
	WRITE:	Block may be shorter than expected. Try Space-Reverse and erasing.

On Read or Write, check length of data transfer to see if any bytes were transferred. Some bytes may be transferred if there was an error condition and the tape drive became not ready during an attempt to reread or rewrite the data.

E4 Write-protect.

E6 Timeout from Halt-I/O or on Read if transport stops after reading a length of blank tape. Timeout from driver based on timeout configuration parameters (refer to Magnetic Tape Configuration Parameter Block).

F3 Channel down. An operation complete interrupt was received but no operation was pending. Something may be wrong with the controller. The F3 message is sent just once per abnormal interrupt received.

CHAPTER 5

PROGRAM LOADER

5.1 GENERAL DESCRIPTION

5.1.1 Function

The program loader is a TRAP #4 server function and is used to:

a. Create a new task.

b. Allocate memory segments for the task based on segment information found in the Loader Information Block (LIB) of a load file created by the linkage editor. The loader forces segments of position independent tasks to be loaded on logical addresses that are multiples of 64Kb.

WARNING

PROGRAMS THAT ARE NOT POSITION-INDEPENDENT BUT ARE LINKED USING ATTRIBUTE P WILL NOT EXECUTE PROPERLY.

c. Read the contents of each segment from the load file into the segments allocated.

5.1.2 Calling Sequence

The task created by the loader is in a DORMANT state following a successful completion of that load function.

The calling sequence is:

```
MOVE.L      #1,D0       Directive number 1 for the loader.
LEA         LOADPB,A0   Parameter block address in A0.
TRAP        #4          Call the TRAP #4 Server.
```

5.1.3 Loader Parameter Block

The loader parameter block consists of three parts:

a. A parameter block used with the Create Task Control Block (CRTCB) directive.

b. Loader options and command line information.

c. An FHS parameter block that defines the load module file containing the contents of the task to be loaded. This part may be omitted if the calling task has assigned the load file and has passed the logical unit number of that file to the loader as part of the loader options.

Table 5-1 describes the first two parts. The third part is described in Appendix A.

TABLE 5-1. Parameter Block

OFFSET	BYTES	CONTENT
$00	4	Taskname (4-byte ASCII name)
$04	4	Session number
$08	2	CRTCB (Create TCB) options
		Bit 15=1 The new task's monitor is specified in the monitor fields that follow.
		Bit 14=1 The new task's monitor is the same as the calling task's monitor.
$0A	4	Monitor taskname
$0E	4	Monitor session number
$12	1	Initial priority If 0, this field is set equal to the calling task's current priority.
$13	1	Limit priority If 0, this field is set equal to the calling task's limit priority.
$14	2	Task attributes
		Bit 15=1 New task is a system task (this bit is ignored if calling task is a user task).
		Bit 13=1 Crash system if new task terminates abnormally (this bit is ignored if new task is a user task).
		Bit 12=1 Dump the contents of memory to a disk file if task terminates abnormally.
$16	4	Task entry address
$1A	2	Task user I.D. If caller is user task, this field is set equal to caller's user I.D.
$1C	2	Load options
		Bit 15=1 Calling task proceeds while new task is loaded.

TABLE 5-1. Parameter Block (cont'd)

OFFSET	BYTES	CONTENT
		Bit 15=0 Calling task waits until the load is complete. Bit 14=1 Loader must assign load file; caller provides an FHS parameter block. Bit 14=0 Caller has assigned load file; a logical unit number is provided. Bit 13=1 A command line, to be passed to the new task, is provided by the caller at the address given. Bit 13=0 No command line is provided by caller. Bits 7-0 Each of these option bits refers to a specific field in the CRTCB portion of the parameter block. If the bit=1, the value supplied in the parameter block is used by the loader. If the bit=0, the loader uses the value found in the LIB of the load file. Bit 7 Logical entry address Bit 6 Task attributes Bit 5 Task limit priority Bit 4 Task initial priority Bit 3 Monitor taskname and session number Bit 2 CRTCB options Bit 1 Task session Bit 0 Taskname
$1E	1	Command line length, in bytes (if bit 13 of loader options = 1)
$1F	1	Logical unit number of load file (if bit 14 of loader options = 0)
$20	8	Reserved
$28	4	Command line address (if bit 13 of loader options = 1)
$2C	34	An FHS parameter block with all the file descriptor fields filled in by the caller. Required field if bit 14 of loader options = 1. Refer to Appendix A.

5

5.2 OPERATION

5.2.1 Loading a User Task

Any task in the system can load a user task. If the task that calls the loader is also a user task, then the session number and user ID of the new task is always set equal to the calling task's session number and user ID. If the calling task is a system task, it can provide a new session number and user ID.

5.2.2 Loading a System Task

A system task is a privileged task and can only be loaded if another system task calls the loader. The new task is loaded as a system task if either set of conditions is met:

a. Bit 6 of the loader options = 1 (use task attributes supplied) and bit 15 of task attributes = 1.

b. Bit 6 of the loader options = 0, bit 15 of the task attributes found in the LIB of the load file = 1, and the user ID supplied by the caller = 0. If these conditions are met, the new task's session number and priority is also be taken from the LIB of the load file.

To generate a system task it must be linked, with attributes set, to the system. It can only be started by user 0 or by another system task.

5.2.3 Return Parameters

When the load is completed, the entire loader parameter block is moved back to the caller's address space with all fields filled in with the values used in the load process. This provides the caller with the information needed to identify and communicate with the new task.

D0 = 0	Successful load.
D0 = 180000XX	An error status was returned as a result of a call to FHS.
D0 = 100000XX	An error status was returned as a result of a call to IOS.
D0 = 20000001	An undefined directive number was provided in D0 on entry.
D0 = 20000012	Load segment logical address not within segment boundaries.
D0 = 20000013	File referenced is not a load file.

A

APPENDIX A
FHS PARAMETER BLOCKS
(TRAP #3)

CODE 0 COMMANDS - DEVICE/FILE COMMANDS

PARAMETER	# OF BYTES	ALLOCATE REQ	ALLOCATE RTN	ASSIGN REQ	ASSIGN RTN	CHANGE ACCESS PERMISSION REQ	CHANGE ACCESS PERMISSION RTN	RENAME REQ	RENAME RTN	PROTECT REQ	PROTECT RTN	CLOSE REQ	CLOSE RTN	DELETE REQ	DELETE RTN	CHECKPOINT REQ	CHECKPOINT RTN
COMMAND	1	$80		$40		$20		$10		$8		$4		$2		$1	
OPTIONS	2	A		A	C	A											
STATUS	1		A		A		A		A		A		A		A		A
LUN	1			A		A		A		A		A				A	
VOLUME	4	D	D	D	D			C						D	D		
USER #	2	D	D	D	D			C						D	D		
CATALOG	8	A		C	C			C						A			
FILENAME	8	A		C	C			C						A			
EXTENSION	2	A		C	C			C						A			
RESERVED	2	$0		$0		$0		$0		$0		$0		$0		$0	
WRITE CODE	1	A		C		C				C				A			
READ CODE	1	A		C		C				C				A			
RECORD LENGTH	2	A			C												
SIZE/POINTER	4	A			C												
SHARED BUFFER START ADDRESS	4			C													
SHARED BUFFER END ADDRESS	4				C												
SHARED BUFFER NAME	4				C												

REQUIRED FIELD:

A = ALWAYS REQUIRED
C = CONDITIONALLY REQUIRED
D = DEFAULT VALUES MAY BE SPECIFIED

RETURNED FIELD:

A = ALWAYS RETURNED
C = CONDITIONALLY RETURNED
D = ACTUAL VALUE RETURNED IF DEFAULT VALUE WAS SPECIFIED

A

FHS PARAMETER BLOCKS TRAP #3

CODE 1 COMMANDS - UTILITY COMMANDS

		RETRIEVE ATTRIBUTES		FETCH DIRECTORY ENTRY		FETCH DEFAULT VOLUME	
PARAMETER	# OF BYTES	REQ	RTN	REQ	RTN	REQ	RTN
COMMAND	1	$80		$40		$8	
OPTIONS	2		A			A	
STATUS	1		A		A		A
LUN	1	A		A			
VOLUME ID	4		A				A
USER NUMBER	2		C	C			
CATALOG	8		C	C			
FILENAME	8		C	C			
EXTENSION	2		C	C			
RESERVED	2	$0		$0		$0	
WRITE CODE	1		A				
READ CODE	1		A				
RECORD LENGTH	2		A				
SIZE/POINTER	4		A	P	P		

REQUIRED FIELD:

A = ALWAYS REQUIRED
C = CONDITIONALLY REQUIRED
P = ALWAYS REQUIRED - POINTER TO CALLER'S ADDRESS SPACE

RETURNED FIELD:

A = ALWAYS RETURNED
C = CONDITIONALLY RETURNED
P = DATA RETURNED AT LOCATION POINTED TO IN CALLER'S ADDRESS SPACE

FHS PARAMETER BLOCKS TRAP #3

CODE 1 COMMANDS - UTILITY COMMANDS

REDEFINED PARAMETER BLOCKS

		FETCH DEVICE MNEMONICS				CHANGE LUN ASSIGNMENT	
PARAMETER	# OF BYTES	REQ	RTN	PARAMETER	# OF BYTES	REQ	RTN
COMMAND	1	$20		COMMAND	1	$10	
OPTIONS	2		A	OPTIONS	2	A	
STATUS	1		A	STATUS	1		A
LUN	1			LUA	1	A	
POINTER	4	P	P	LUB	1	A	
LENGTH	4	A	A	RESERVED	1	$0	
				TASKNAME	4	A	
				TASK SESSION	4	A	

REQUIRED FIELD:

A = ALWAYS REQUIRED
P = ALWAYS REQUIRED - POINTER TO CALLER'S ADDRESS SPACE

RETURNED FIELD:

A = ALWAYS RETURNED
P = DATA RETURNED IN CALLER'S ADDRESS SPACE

FHS PARAMETER BLOCKS TRAP #3

CODE 2 COMMANDS - SPOOLER TASK COMMANDS

		CANCEL		CONTINUE		FORMS		PRINT		COPIES		QUEUE	
PARAMETER	# OF BYTES	REQ	RTN	REQ	RTN	REQ	RTN	REQ	RTN	REQ	RTN	REQ	RTN
COMMAND	1	$80		$40		$20		$10		$8		$4	
OPTIONS	2							A		A		A	
STATUS	1		A		A		A		A		A		A
LUN	1												
VOLUME ID	4	C		A		A		C		C		A	
USER NUMBER	2	A						A		A		A	
CATALOG	8	C						C		C		U	
FILENAME	8	C						C		C			
EXTENSION	2	C						C		C			
RESERVED	2												
RESERVED	4												
JOB ID	4	C				F		C		C			
DEVICE NAME	4							C					
FORMS ID	4							C					

REQUIRED FIELD:

A = ALWAYS REQUIRED
C = CONDITIONALLY REQUIRED
F = ALWAYS REQUIRED - REDEFINED AS FORMS ID
U = ALWAYS REQUIRED - REDEFINED AS USER'S CONSOLE NAME

RETURNED FIELD:

A = ALWAYS RETURNED

THIS PAGE INTENTIONALLY LEFT BLANK.

APPENDIX B
IOS PARAMETER BLOCKS
(TRAP #2)

IOS PARAMETER BLOCKS TRAP #2

CODE 0 COMMANDS

DATA TRANSFER FUNCTION

PARAMETER	# OF BYTES	READ REQ	READ RTN	WRITE REQ	WRITE RTN	OUTPUT WITH INPUT REQ	OUTPUT WITH INPUT RTN	UPDATE RECORD REQ	UPDATE RECORD RTN	DELETE RECORD REQ	DELETE RECORD RTN	FORMAT DISK REQ	FORMAT DISK RTN	TRANSMIT BREAK REQ	TRANSMIT BREAK RTN
CODE	1	$0		$0		$0		$0		$0		$0		$0	
FUNCTION	1	$1		$2		$4		$8		$10		$20		$40	
OPTIONS	2	A		A		A		A		A		A		A	
STATUS	1		A		A		A		A		A		A		A
LUN	1	A		A		A		A		A		A		A	
RESERVED	2	$0		$0		$0		$0		$0		$0		$10	
RANDOM RECORD NUMBER	4	C	C	C	C	A		C	A	C		C			
BUFFER START ADDRESS	4	A		A		A		A		C		C			
BUFFER END ADDRESS	4	A		A		A		A		C					
LENGTH OF DATA TRANSFER	4		A		A	A			A						
COMPLETION SERVICE ADDRESS	4	C		C		C		C		C					

REQUIRED FIELD:

A = ALWAYS REQUIRED
C = CONDITIONALLY REQUIRED

RETURNED FIELD:

A = ALWAYS RETURNED
C = CONDITIONALLY RETURNED

IOS PARAMETER BLOCKS TRAP #2

B

CODE 1 COMMAND FUNCTIONS

PARAMETER	# OF BYTES	POSITION REQ	POSITION RTN	REWIND REQ	REWIND RTN	LOCAL BREAK CLAIMER REQ	LOCAL BREAK CLAIMER RTN
CODE	1	$01		$01		$01	
FUNCTION	1	$01		$02		$20	
OPTIONS	2	A		A		C	
STATUS	1		A		A		A
LUN	1	A		A		A	
RESERVED	2	$0		$0		$0	
RANDOM RECORD NUMBER	4	A	A		A		
BUFFER START ADDRESS	4						
BUFFER END ADDRESS	4						
LENGTH OF DATA TRANSFER	4						
COMPLETION SERVICE ADDRESS	4					C	

REQUIRED FIELD:

A = ALWAYS REQUIRED
C = CONDITIONALLY REQUIRED

RETURNED FIELD:

A = ALWAYS RETURNED
C = CONDITIONALLY RETURNED

IOS PARAMETER BLOCKS TRAP #2

B

CODE 1 COMMAND FUNCTIONS

PARAMETER	# OF BYTES	TEST I/O COMPLETE REQ	TEST I/O COMPLETE RTN	WAIT ONLY REQ	WAIT ONLY RTN	HALT I/O REQ	HALT I/O RTN
CODE	1	$01		$01		$01	
FUNCTION	1	$04		$08		$10	
OPTIONS	2						
STATUS	1		Z		C		B
LUN	1	A		A		A	
RESERVED	2	$0		$0		$0	

REQUIRED FIELD:

A = ALWAYS REQUIRED

RETURNED FIELD:

B = STATUS RETURNED IN BOTH I/O PARAMETER BLOCKS

A = ALWAYS RETURNED

Z = Z BIT ALWAYS RETURNED
1 = NO I/O 0 = I/O EXISTS

C = ALWAYS RETURNED TO ORIGINAL I/O PROCEED PARAMETER BLOCK

IOS PARAMETER BLOCKS TRAP #2

B

CODE 1 COMMAND FUNCTIONS

PARAMETER	# OF BYTES	CONFIGURATION-STATUS REQUEST REQ	RTN	CONFIGURE-DEVICE REQUEST REQ	RTN
CODE	1	$01		$01	
FUNCTION	1	$40		$80	
OPTIONS	2				
STATUS	1		AZ		AZ
LUN	1	A		A	
RESERVED	2	$00		$00	

REQUIRED FIELD:

A = ALWAYS REQUIRED

RETURNED FIELD:

A = ALWAYS RETURNED

Z = Z BIT ALWAYS RETURNED

0 = REQUEST ACCEPTED/SUCCESSFUL

1 = REQUEST REJECTED/UNSUCCESSFUL

CODE 2 COMMAND FUNCTIONS

PARAMETER	# OF BYTES	NEGATE LOCAL BREAK CLAIMER REQ	RTN
CODE	1	$02	
FUNCTION	1	$01	
OPTIONS	2		
STATUS	1		A,Z
LUN	1	A	
RESERVED	2	$00	

REQUIRED FIELD:

A = ALWAYS REQUIRED

RETURNED FIELD:

A = ALWAYS RETURNED

Z = Z BIT ALWAYS RETURNED

0 = REQUEST ACCEPTED/SUCCESSFUL

1 = REQUEST REJECTED/UNSUCCESSFUL

IOS PARAMETER BLOCKS TRAP #2

B

CODE 2 COMMAND FUNCTIONS

CLAIM DRIVER EVENTS

PARAMETER	# OF BYTES	REQ	RTN
CODE	1	$02	
FUNCTION	1	$02	
OPTIONS	2		
STATUS	1		A,Z
LUN	1	A	
RESERVED	2		

REQUIRED FIELD:

A = ALWAYS REQUIRED

RETURNED FIELD:

A = ALWAYS RETURNED

Z = Z BIT ALWAYS RETURNED

0 = REQUEST ACCEPTED/SUCCESSFUL

1 = REQUEST REJECTED/UNSUCCESSFUL

IOS PARAMETER BLOCKS TRAP #2

B

CODE 2 COMMAND FUNCTIONS

NEGATE DRIVER EVENTS

PARAMETER	# OF BYTES	REQ	RTN
CODE	1	$02	
FUNCTION	1	$04	
OPTIONS	2		
STATUS	1		A,Z
LUN	1	A	
RESERVED	2		

REQUIRED FIELD:

A = ALWAYS REQUIRED

RETURNED FIELD:

A = ALWAYS RETURNED

Z = Z BIT ALWAYS RETURNED

0 = REQUEST ACCEPTED/SUCCESSFUL

1 = REQUEST REJECTED/UNSUCCESSFUL

IOS PARAMETER BLOCKS TRAP #2

B

CODE 80 COMMAND FUNCTIONS

CONFIGURE DEFAULTS
REQUEST

PARAMETER	BYTES	REQ	# OF RTN
CODE	1	$80	
FUNCTION	1	$02	
OPTIONS	2		
STATUS	1		AZ
LUN	1	A	
RESERVED	2	$00	

REQUIRED FIELD:

A = ALWAYS REQUIRED

RETURNED FIELD:

A = ALWAYS RETURNED

Z = Z BIT ALWAYS RETURNED

0 = REQUEST ACCEPTED/SUCCESSFUL

1 = REQUEST REJECTED/UNSUCCESSFUL

IOS PARAMETER BLOCK TRAP #2

CODE $80 COMMAND FUNCTION

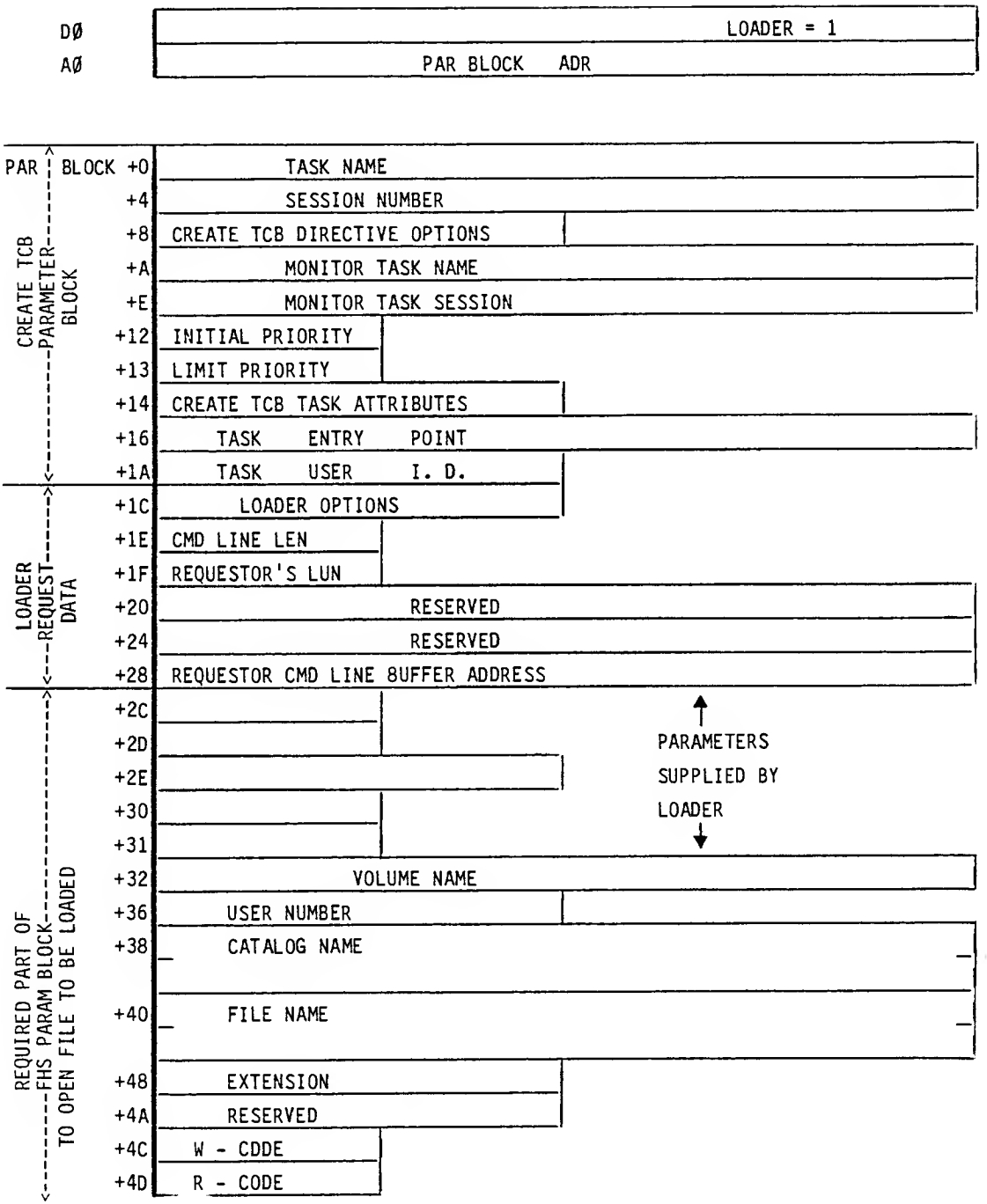
PARAMETER	# OF BYTES	UNCLAIMED BREAKS (SYSTEM FUNCTIONS) REQ	RTN
CODE	1	$80	
FUNCTION	1	$01	
OPTIONS	2		
STATUS	1		A
LUN	1		
RESERVED	2	$0	

RETURNED FIELD:

A = ALWAYS RETURNED

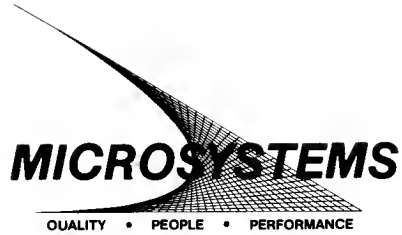
APPENDIX C
LOADER PARAMETER BLOCKS
(TRAP #4)

LOADER PARAMETER BLOCKS TRAP #4



INDEX

SUGGESTION/PROBLEM REPORT



Motorola welcomes your comments on its products and publications. Please use this form.

To: Motorola Inc.
Microsystems
2900 S. Diablo Way
Tempe, Arizona 85282
Attention: Publications Manager
Maildrop DW164

Product: _____ Manual: _____

COMMENTS: _____

_____

_____

_____

_____

_____

_____

_____

_____

_____

_____

_____

Please Print

Name _____ Title _____

Company _____ Division _____

Street _____ Mail Drop _____ Phone _____

City _____ State _____ Zip _____

For Additional Motorola Publications
Literature Distribution Center
616 West 24th Street
Tempe, AZ 85282
(602) 994-6561

Four Phase/Motorola Customer Support, Tempe Operations
(800) 528-1908
(602) 438-3100

MOTOROLA

MOTOROLA ***Semiconductor Products Inc.***

P.O. BOX 20912 • PHOENIX, ARIZONA 85036 • A SUBSIDIARY OF MOTOROLA INC.

19144 PRINTED IN USA (1/86) MESSENGER 3500